MODEL－1883

Universal PROM プログラマ
オペレーション　マニュアル

ミナトエレクトロニクス（株）
第1版　2012．11
第2版　2013．03

M1883JP-100B 201303.doc

MODEL1883 プログラマ　梱包一覧

MODEL1883(以下、M1883)をお買い上げ時に同梱されているものを記載しています。
開封時に内容及び数量の確認をお願いします。
万一、異常がありましたら、販売店または、弊社までご連絡ください。

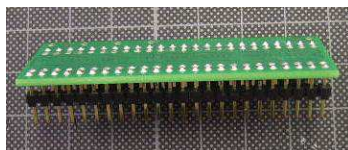
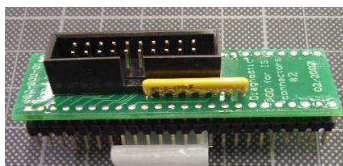
M1883プログラマ

CD－ROM

・M1883　セットアップソフト
　（M1883コントロールソフト　及び　USBドライバーのインストール）
・M1883　オペレーション　マニュアル

電源ケーブル（日本国内規格品）

USBケーブル

ZIFソケット検査用POD1

ISPコネクタ検査用POD2

ISPケーブル

（ISPチェック用）

安全にお使い頂く為に

安全上の注意

このユーザーズマニュアルには、Ｍ１８８３を安全に正しくお使い頂く為に安全表示が記述されています。
Ｍ１８８３を安全に正しくお使い頂いて、お使いになる方や他の人々への危害や財産への損害を未然に防止する為に、次のように絵表示で説明しています。
これら絵表示と意味を十分理解した上で本書をお読みください。また本書は保管して、必要に応じて参照してください。

絵表示の説明

警告	この表示の注意事項を守らないと、使用者が死亡または重傷を負う可能性が想定される内容を示しています。
注意	この表示の注意事項を守らないと、使用者の怪我または物的損害の発生が考えられる内容を示しています。

警告

強制

本製品を使用する際は、必ず弊社ミナトエレクトロニクス(株)が提示する警告、注意指示に従ってください。

分解禁止

本製品の分解や改造はしないでください。火災や感電のおそれがあります。

電源プラグを抜く

煙が出たり、変な臭いや音がしたら、すぐに AC コンセントからプラグを抜いてください。
そのまま使用を続けると、ショートにより火災や感電する恐れがあります。

電源プラグを抜く

本製品を落としたり、強い衝撃を与えたりした場合は、すぐにACコンセントから電源プラグを抜いてください。

そのまま使用を続けると、ショートにより火災や感電する恐れがあります。
弊社修理窓口にご相談ください。

電源プラグを抜く

液体や異物などが内部に入ったら、すぐにACコンセントから電源プラグを抜いてください。

そのまま使用を続けると、ショートにより火災や感電する恐れがあります。

弊社修理窓口にご相談ください。

⚠注意

強制

本製品ご使用の際は、本取扱説明書をご理解されたM1883のオペレータの方が操作に当たってください。

誤ったオペレーションは、本製品またはデバイスを破損させる可能性があります。

強制

静電気による破損を防ぐために、本製品にふれる前に身近な広い面積の金属に素手で触れて、身体の静電気を取り除くようにしてください。

静電気により、本製品またはデバイスを破損する恐れがあります。

強制

本体表面、デバイスソケットの清掃をしてください。

ほこりがたまったままのご使用は、火災や故障の原因になることがあります。定期的な清掃をしてください。

強制

PASS ／ FAILの判定はプログラマ本体のGOOD、ERROR　LED以外にパソコン画面上のチェックサムを確認してください。

パソコンの誤操作等によりバッファ内容を変更している場合があります。ですから定期的に（少なくともし作業開始時と作業終了時）チェックサムを確認してください。チェックサムの確認を怠ると書き込み不良のデバイスが製品に混入する恐れがあります。

強制

付属品についての注意事項
本製品に付属の電源ケーブルは日本国内専用のケーブルです。海外で使用する場合は、その国の電源規格に適合した電源ケーブルが必要です。

目 次

第２章　Ｍ１８８３コントロールソフト　操作マニュアル

ご使用の前に

お客様各位

ミナトエレクトロニクス製　MODEL1883　ユニバーサルプログラマ（略称：M1883プログラマ）をお買い上げ、ありがとうございます。本製品の保証期間は、納入後１年とさせて頂きます。但し、保証期間内においても、天災による損傷、ご使用上の操作ミス、お客様による改造・変更、またデバイスソケットの消耗に対する保証は致しかねます。
尚、本機を御使用する事で発生した直接的、間接的トラブルに関して、ミナトエレクトロニクス㈱は一切の責任を負いかねます。詳細内容は本マニュアルの巻末「保障規定」を参照してください。ご不明の点は、弊社サービスまたは各営業所にご連絡ください。

このマニュアルの著作権について

1. 本マニュアルは、お客様が用途に応じて適切なデバイス書き込みをしていただくための資料であり、本マニュアル中に記載の技術情報についてミナトエレクトロニクスが所有する知的財産権その他の権利の実施、使用を許諾するものではありません。

2. 本マニュアルに記載の製品データ、図、表、プログラム、アルゴリズムその他応用回路例の使用に起因する損害、第三者所有の権利に対する侵害に関し、ミナトエレクトロニクスは責任を負いません。

3. 本マニュアルに記載の製品データ、図、表、プログラム、アルゴリズムその他全ての情報は本マニュア発行時点のものであり、ミナトエレクトロニクスは、予告なしに、記載内容または仕様を変更することがあります。
最新のデバイス対応アルゴリズムを開発、更新しておりますのでミナトエレクトロニクスホームページ（http://www.minato.co.jp）などを通じて公開される情報をご活用ください。

4. 本マニュアルに記載した情報は、正確を期すため、慎重に制作したものですが万一記述誤りに起因する損害がお客様に生じた場合には、ミナトエレクトロニクスはその責任を負いません。

5. 本マニュアルの転載、複製については、文書によるミナトエレクトロニクスの事前の承諾が必要です。

6. 本マニュアルに関し詳細についてのお問い合わせ、その他お気付きの点がございましたら
ミナトエレクトロニクス、または販売代理店までご照会ください。

マニュアルの使用方法

このマニュアルはコントロールソフトのインストール方法と、M1883プログラマの使用方法について、解説しています。マニュアルの解説内容は「パソコン操作」と「ソフトのインストールの経験」がある人を対象に解説しています。
コントロールソフトをパソコンにインストールした後は、コントロールソフト上の‘ヘルプ’メニューからも、このマニュアルと同等の解説を参照することが出来ます。
また最新版のマニュアルをホームページでも公開していますので、ご利用ください。

マニュアル上の表記

M1883プログラマ用コントロールソフトで使用するファンクション名は太字で表記しています。

ファイル–>読込、**ファイル–>保存**、**バッファ–>表示・編集**、

ツールボタンは

Load、**Save**、**Edit**、**Select**、. . . .

またキーボード上で使用するファンクションキーは

<F1>、**<F5>**、**etc.**

で表記しています。

このマニュアル上で使用されている用語の解説

Device　：ユーザーデータを書き込むことが可能なデバイス(IC)。

ZIF socket　：M1883プログラマの上面についているソケットのこと。
書き込むデバイスをこのソケットに実装する
(ZIF：Zero insertion Forceの略)。

バッファ　：コントロールソフトがパソコン上に一時的に用意したメモリのこと。
マスターROMのリードまたは、データファイルの読込データがバッファメモリに一時的に記録されます。ファイル保存時にはバッファメモリ内のデータがファイルに保存されます。

Printer port　：パソコンのプリンターポート
M1883プログラマをパラレルケーブルで接続するポート。

USB port　：パソコンのUSBポート
M1883プログラマをUSBケーブルで接続するポート。

HEX data format
：M1883プログラマがリード可能なデータファイル形式のひとつで、コントロールソフトの表示/編集機能で内容が確認できる。
一般的に使用されるHEXフォーマットとして
INTEL. hex
Motorola. mot
が使用されています。

第1章

プログラマ概要、仕様、インストール

M1883プログラマ　概要

M1883プログラマはUSBインターフェイス、パラレルインターフェイスと48ピン分の高機能ピンドライバー回路を持った高速のユニバーサルタイプのプログラマです。
またISP（インシステム・プログラム）コネクタも装備していますので、部品実装済みのボード上でも書込みが出来ます。

M1883プログラマはIBM－PC互換機でUSBポートまたはプリンターポートを持ったパソコンならば、ほとんど全てのパソコンで使用可能です。

またM1883プログラマ本体には汎用性の高い高機能ピンドライバー、広い電圧範囲をカバーする電源ピン、分解能の高いD／A回路を採用していますので、今後サポートされるデバイスに柔軟に対応することが可能です。

またコントロールソフトはプルダウンメニュー、ホットキー、ヘルプ機能をサポートしていますので、簡単に操作することが出来ます。

製品の構成

コントロールソフトをインストールする前およびプログラマを使用する前に、今回お届けした製品の梱包内容を確認してください。

不足品や故障しているパーツがある場合はすぐに代理店または営業所に連絡してください。

梱包内容一覧

（1） M1883プログラマ本体	1台
（2） 電源ケーブル	1本
（3） USBケーブル（1．5m）	1本
（4） ZIFソケットのセルフチェック用POD	1個
（5） ISPコネクタのセルフチェック用POD	1個
（6） ISPチェック用フラットケーブル	1本
（7） M1883用セットアップソフト入りCD－ROM （PDF版オペレーション　マニュアル内蔵）	1枚

注意　：　付属の電源ケーブルは日本国内専用のケーブルです。海外で使用する場合は、その国の電源規格にあった電源ケーブルが必要です。

Ｍ１８８３プログラマに接続するパソコン環境

最低限のパソコン環境

・OS	Microsoft Windows® 2000 以降
・CPU	ペンティアム４以上
・RAM容量	５１２MB以上
・ハードディスク	２００MB以上の空き容量を持ったDISK
・インターフェイス	USBポートVer１．１以降（プログラマをUSBポートで接続時） またはプリンターポート（プログラマをLPTポートで接続時）
・CD ドライブ	CD－ROMリーダー

推奨するパソコン環境

・OS	Microsoft Windows® 7 (32-bit または64-bit)
・CPU	Core ２ Duo
・RAM容量	１GB以上
・ハードディスク	１GB以上の空き容量を持ったDISK
・インターフェイス	USBポートVer２．０以降（プログラマをUSBポートで接続時） またはプリンターポート：ECP、EPPモード（プログラマをLPTポートで接続時）
・CD ドライブ	CD－ROMリーダー

ハードディスクの空き容量はお客様が使用するターゲットデバイスの容量に依存します。
大容量のデバイスを書き込むためには1000MB ＋ デバイスサイズの2倍以上のハードディスク容量が必要になる場合があります。大容量デバイスをスムーズに書き込むためには、空き容量が十分なハードディスクを使用してください。

M1883プログラマの特徴

M1883はUSB／LPTポートを持ったWindowsベースのパソコンで利用できる、次世代のユニバーサルプログラマです。このプログラマは小ロット生産や回路設計技術者向けに設計された、高信頼性かつ高速動作するプログラマです。

M1883はデバイスのパッケージに合わせた変換アダプタを使用することで、全てのプログラマブル・デバイスをサポートします。
技術者は回路設計に最適なデバイスを自由に選択することが出来ます。
またON＿BOARD書き込み用のISP（インシステム・プログラミング）コネクタを使用して基板上のデバイスに直接データを書き込むことが出来ます。

M1883はプログラマの機能以外にTTL／CMOSロジックやメモリICをテストする機能も持っています。さらにそのテスト機能ではユーザーが定義したテストパターンも利用出来ます。

M1883は高い信頼性を持った高機能なプログラマです。このクラスのプログラマのなかでは他社と比べもっともコストパフォーマンスにすぐれています。

M1883は高速FPGAドライバー回路を採用しています。また最適化されたアルゴリズムを採用していますので、非常に速い書き込み環境を提供します。
高速書き込みが出来ますので、小ロット生産にも適したプログラマです。

M1883はIBM＿PC互換パソコンでインターフェイスにUSB(2.0/1.1)またはプリンターポートを持っていれば接続可能です。
USBとパラレルの両方をサポートしていますので、ユーザーが使用するパソコンに合わせて、どちらか一方を選択出来ます。
最新のノートパソコンでも、古くなったデスクトップパソコンでも使用出来ます。

M1883は静電気対策用リストバンドを接続するためのバナナジャック端子を装備しています。
またアース接続用のバナナジャック端子も装備していますので簡単に静電気対策が出来ます。

M1883はFPGAベースの高機能な48ピンTTLドライバー回路を持っています。このドライバー回路は2．0vまでのあらゆるデバイスに対応できる高信頼性と高速動作を兼ね備えています。

M1883はターゲットデバイスにデータを書き込む前に誤挿入チェックとコンタクトチェック（デバイスとソケット間の接触）を行なっています。
さらに過電流保護とデバイスIDチェック機能もありますので、オペレータの作業ミスによるデバイスへのダメージを未然に防止することが出来ます。

M1883は電気回路の保護機能がありますので、作業者や周辺で発生する静電気からプログラマを保護します。
プログラマの全ての入力回路にこの保護回路がついています。ZIFソケット、ISPコネクタ、インターフェイス回路、電源入力部分を保護しています。

M1883ではVccマージンベリファイを行なっていますので、高い書き込みの信頼性を得ることが出来ます。
また各種のソケットアダプタを用意していますので、PLCC、SOIC、TSOP、QFP、SDIP、BGA等多種類のパッケージに対応しています。

M1883用コントロールソフトはプルダウンメニューやショートカットキーやオンラインヘルプ機能を持っていますので、簡単に操作することが出来ます。
デバイス選択はメーカー名、デバイス名、デバイスの種類指定から簡単に選択出来ます。
デバイス用コマンド（Read, Blank, Program, Verify, Erase）ではチェック機能（誤挿入チェック、シグネチャーチェック）、スペシャル機能（自動番号付けモード、自動書き込みモード）を利用出来ます。

M1883用コントロールソフトでサポートしているデータフォーマットは、自動でファイルフォーマットを判別し、データをパソコンにロード出来ます。

M1883用コントロールソフトのシリアライズ機能（シリアル番号管理機能）は個々のデバイスに
それぞれ異なったシリアル番号を割り当てることが出来ます。また単純に増加するシリアル番号を
つけることも出来ます。

M1883用コントロールソフトは多くのデバイス情報を提供します。
たとえば利用可能なパッケージ寸法やデバイスの捺印情報を提供しています。
ISPを使用するための多くの情報も提供します。
選択したデバイスで使用するISPコネクタのピン情報やISPで書き込むために必要な回路構成も詳細に説明しています。

M1883用コントロールソフトは他のアプリケーションからBATファイル・コマンドやDLLファイルを使用してリモート・コントロール制御が可能です。
DLLはC言語やVBASICで供給されている標準的なパーツです。
JEDEC標準のJam filesはJamPlayerによって解釈／実行されます。
VMEファイルはVME Playerによって解釈／実行されます。
Jam filesおよびVME filesは各々のデバイスメーカーが開発し、提供しています。

新しいデバイスのサポート

多くの場合新しいデバイスはM1883プログラマのソフトウェアのアップデートだけで対応
出来ます。汎用タイプのユニバーサルプログラマですので、コントロールソフトのバージョンアップだけで、新しいデバイスに対応出来ます。

プログラマのアップデートについて

なぜ最新版のデバイス書き込み仕様(書き込みアルゴリズム)が重要なのか?

半導体メーカーは新しい技術、新しい製造プロセス、新しいパッケージのデバイスを絶え間なく開発しユーザーに供給しています。
またこれらのデバイスを書き込むために、書き込み装置メーカーにも多くのデバイス情報を送ってきています。われわれはこれらの半導体メーカーの開発テンポに合わせるように、年間を通してデバイスアルゴリズムを開発し、サポートしています。

半導体メーカーは新製品だけを開発しているわけではありません。新製品開発時にできた新しい技術を、既存のデバイスにも反映させています。これは製造コストの削減や、デバイスの性能向上等のために行なわれています。
これらの変更は時には「**書き込みアルゴリズムに影響をあたえる**」ことがあります。
多くの場合、古いアルゴリズムでデバイスをプログラムした場合、それらは最新のアルゴリズムでプログラムした場合とくらべ、十分な性能が保障されないかもしれません。最新のアルゴリズムを使用しない場合、書き込みの歩留まり低下、書き込み時間の増加、さらに長期でみた場合のデバイスの信頼性に影響をあたえる可能性があります。

最新の書き込みアルゴリズムを使用することは、書き込みの信頼性を上げる重要なキーになります。

新しいデバイスが開発された時やデバイスの書き込みに変更があった場合には、最新のアルゴリズムを使用して書き込みができるように、アルゴリズムの更新と作成を随時行なっています。

M1883コントロールソフトをアップデートすることにより、アップデート時点の最新アルゴリズムに更新することが出来ます。
コントロールソフトの最新版へのアップデートは、代理店や営業所に問い合わせください。

クイックスタート

ソフトウェアのインストール

付属のCD－ROMをパソコンにセットし、“pg4uwarm．exe”を実行してください。
表示画面の指示に従って操作してください。
（詳細は23ページのインストールを参照ください。）

プログラマとハードウェアの接続

M1883プログラマのUSBポートとパソコンのUSBポートを付属のUSBケーブルで接続します。
M1883プログラマを付属の電源ケーブルでAC100v（100v～240v）に接続します。
その後プログラマの電源スイッチを入れてください。
プログラマがUSBを使用した新しいハードウェアとしてパソコンにインストールされます。
※重要：**USBケーブル**はソフトウェアのインストール終了後に接続してください。

コントロールソフトの起動

デスクトップ画面上の をダブルクリックしてください。
コントロールソフトはパソコンに接続されているプログラマを検出します。

日本語表示に変更する

TOPメニューのオプション　―＞　オプション設定．．．内の
・言　語　－＞選択する言語　－＞**“Japanese”**
・言　語　－＞選択するヘルプファイル　－＞**“Japanese”**
を選択してください。
コントロールソフトを一度終了してから、コントロールソフトを再起動してください。
日本語に変更されます。

下記はコントロールソフトのTOPメニューです。

ファイル　：データファイルの読込／保存、プロジェクトファイルの読込／保存、コントロールソフトの終了、プログラマの動作状態を保存して終了　等のファイル操作に使用します。

バッファ　：バッファ操作、ブロック操作、データ移動、データ初期化、チェクサム等のデータ編集と表示に使用します。

デバイス　：プログラマブル・デバイスの選択、
Read、Blank、Program、Verify、Erase、書き込み条件変更に使用します。

プログラマ　：M1883プログラマの検出、自動書き込みモード設定、自己診断に使用します。

オプション　：各種ディフォルト設定の確認や変更のために使用します。

ヘルプ　：サポートされているデバイス一覧、プログラマのバージョン表示にに使用します。

デバイス書込み方法（Programming）

（1）使用するデバイス選択を選択します—＞ Select をクリック Select

（2）書き込むデータをバッファに読み込む

a）ファイルからロードする場合 ——＞ Load をクリック Load

b）デバイスからリードする場合 ——＞ZIFソケットにデバイスを装着し、
Read をクリック Read

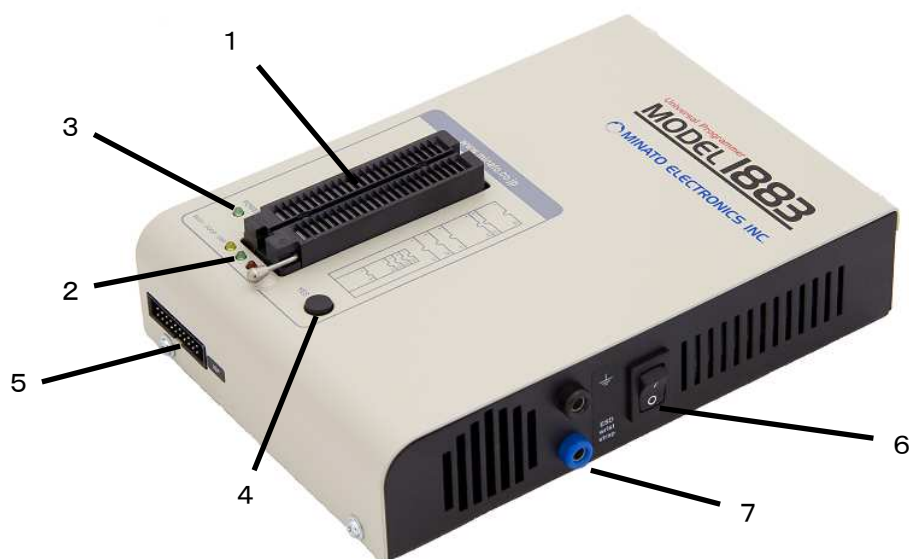
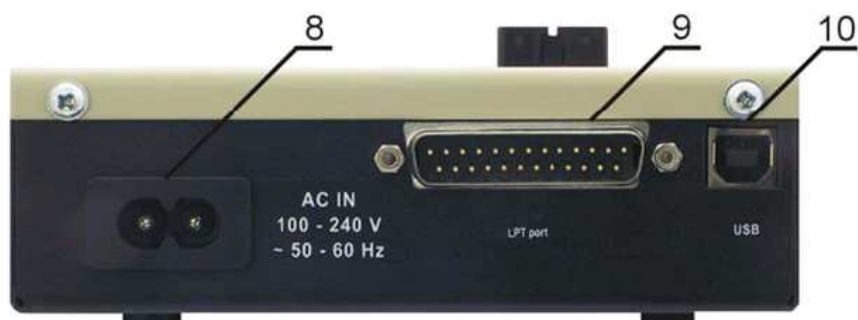
（3）書き込みたいデバイスをZIFソケットに装着

（4）未書き込み状態かチェックします——＞ Blank をクリック Blank

（5）デバイスにデータを書込む ———＞ Program をクリック Program

（6）書き込んだデータとバッファのデータを照合する
———＞ Verify をクリック Verify

M1883プログラマ各部の名称

（1） 48ピン ZIFソケット
（2） GOOD／ERROR（PASS／FAIL） LED（書き込み結果表示）
（3） POWER ON LED
（4） YES！ ボタン（2回目以降のスタートボタン）
（5） ISPコネクタ
（6） 電源スイッチ
（7） アース端子と静電気対策用リストバンド接続端子

（8） 電源ケーブル接続コネクタ（AC100v――AC240v対応）
注意：付属の電源ケーブルは日本国内専用のケーブルです。海外で使用する場合は、その国の電源規格にあった電源ケーブルが必要です。
（9） LPTコネクタ（パソコンのプリンターポート接続用）
（10） USBコネクタ（パソコンのUSBポート接続用）

M1883プログラマとパソコンの接続方法

USBポート使用時

M1883プログラマ本体のUSBコネクタとパソコンのUSBコネクタを付属のUSBケーブルで接続します。M1883プログラマ本体の電源コネクタに付属の電源ケーブルを接続し、AC100v（100v～240v）を接続します。
USBケーブルとACケーブルの接続する順番はどちらが先でもかまいません。
電源をONにしたままでケーブルの接続、取り外しが可能です。

パラレル・ポート使用時

パソコンとM1883プログラマの電源SWをOFFしてください。パソコンの空いているプリンターポートとM1883プログラマのLPTコネクタをパラレルケーブルで接続してください。
（パラレルケーブルはM1883のオプションです）接続後コネクタ部分のビスでしっかり固定してください。

M1883プログラマの接続に機械的な構造のプリンター切り替え器を接続しないでください。
切り替え器が必要な場合はロジック回路方式の切り替え器を使用してください。

パソコンの電源をONしてください。
M1883プログラマ本体の電源コネクタに付属の電源ケーブルを接続し、AC100v（AC100～240v）を接続します。
M1883プログラマの電源SWをONしてください。電源が入ると、M1883プログラマ上のPOWER LEDがやや暗く点灯します。このLEDはプログラマの準備が完了したことを示しています。
次にパソコン上のコントロールソフトを起動します。

プログラマ接続時にパソコンの電源をOFFしたくない場合、次の手順に従ってください。

・プログラマとPCを接続する場合
最初にパラレルケーブルを接続します。その後プログラマの電源ケーブルを接続します。

・プログラマとPCの接続ケーブルをはずす場合
最初にプログラマの電源ケーブルをはずします。その後パラレルケーブルをはずしてください。

M1883プログラマ側だけならば、ケーブルの抜き差しのタイミングはなんら問題になりません
（プログラマ側には保護回路がついています）。
しかしパソコン側で問題が発生する場合がありますので上記の注意事項に従ってください。

デバイスの書き込み手順

（1）書き込むデバイスを選択します。

（2）その後ZIFソケットにデバイスを挿入し、ZIFソケットレバーを倒し、デバイスを固定します。
デバイスの正しい挿入位置はZIFソケット横の図を参照してください。
BUSY　LEDが消えているときにデバイスを抜き差ししてください。

（3）コントロールソフト上のツールバー Program をクリックし、書き込みを開始します。
（2回目以降の書き込み作業時にはM1883プログラマ　ZIFソケット横の“YES”ボタンを押すと
書き込みを開始します）。

BUSY　LEDが点灯し、動作状態を示すプログレスバーが画面に表示されます。

（4）書き込み終了
正常終了時　　――＞GOOD　LED（緑色）が点灯します。
書き込みエラー　――＞ERROR　LED（赤色）が点灯します。

（5）デバイスを交換します。
上記の（3）～（5）の操作を繰り返します。

注意：
書き込み中にターゲットデバイスをソケットから抜いてはいけません。（動作中にはBUSY　LEDが
点灯しています）。
また書き込み中に、コントロールソフトを中断したり、パソコンの電源をOFFしないでください。
中断した場合（瞬間停電や長い停電も含む）、書き込み中のデバイスのデータは不完全な状態のま
まですので、再度消去と再書込みを行ってください。

プログラマ動作中の停電やパソコンの電源OFFは、ユーザーが選択した書き込みパラメータが保
存されていない場合がありますので、デバイスの再設定から行ってください。

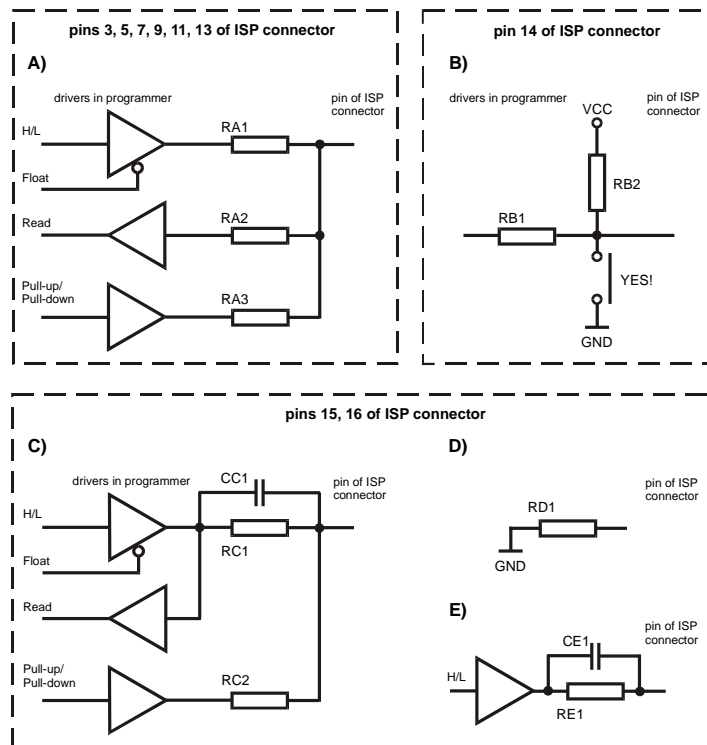
M1883による In-System プログラミング

ISPコネクタの各PINの使用方法は書き込むデバイス毎に異なります。また各デバイスのISP情報を見るには、はじめに使用するデバイスを選択する必要があります。
ISP対応デバイスには選択デバイス名のあとに“ISP”の表示が付いています。
デバイス選択後コントロールソフトのメニューで **デバイス->デバイス情報** でISP情報を表示することが出来ます。
これらの仕様はデバイスメーカーが発表しているアプリケーション・ノートに対応しています。

ISPコネクタ、ケーブル

プログラマ内で使用しているコネクタは TE社製 角型コネクタ 型名：２－１６３４６８９－０ です。

ISPコネクタの正面図

ISPコネクタに接続しているプログラマ内の回路

RA1: 180Ω, RA2:1kΩ, RA3 22kΩ,
RB1: 10kΩ, RB2: 10kΩ,
CC1: 1000pF, RC1: 1kΩ, RC2: 22kΩ,
RD1: 22kΩ, CE1: 1000pF, RE1: 1kΩ,

図−C) 15ピン、16ピンがISPプログラム設定でロジック信号に設定された場合の内部回路
図−D) 図−E) 15ピン、16ピンが LED OK と LED ERROR信号に設定された場合の内部回路
図−D)　ISP動作前のLED回路状態
図−E)　ISP動作後のLED回路状態

Note:
LED OK(点灯）、LED ERROR(点灯)時の信号レベルはHighです。Highレベルは1.8v−−5vですが、ターゲットに選択したISPデバイスの電源電圧に依存します。
LED OK(消灯）、LED ERROR(消灯)時の信号レベルはLowです。Lowレベルは 0v−−0.4vです。
上の回路と抵抗値を考慮してターゲット・システム側の回路を決定して下さい。

ISPコネクタの各PINに対応する信号名、機能は選択したデバイスによって異なります。
デバイス選択後の<u>デバイス−>デバイス情報</u>で確認出来ます。
またISP機能によるプログラムを使用する時は、ISP対応のデバイスを選択して下さい。
デバイス名の後ろに“(ISP)”が記入されているデバイスを選択して下さい。

ISPケーブル:ISPケーブル上のピン番号1にはコネクタ部分に三角マークがついています。
ISPケーブルで使用しているコネクタは
Harting社製　BOXコネクタ　型名:09185207813　です。

警告:
M1883プログラマのISP書き込みを使用する場合、ZIFソケットにデバイスを入れないでください。
ZIFソケットを使用して書き込む場合、ISPケーブルは接続しないでください。
ターゲットデバイスとISPコネクタを接続するケーブルは出来るだけ短くしてください。
ケーブルが長すぎるとうまく動作しない場合があります。

M1883はプログラムするデバイスとターゲット・システムに電源を供給することが出来ます。
プログラムするデバイス用電源　　　　　　:ISPコネクタの1番ピンに出力
ターゲット・システム用電源電流制限付き　　:ISPコネクタの19、20番ピンに出力

M1883はデバイスにプログラミング電圧を印加し、そして、その電圧値をチェックします。
もし、プログラミング電圧が期待したものと異なる場合は、デバイスの書き込みは行ないません。

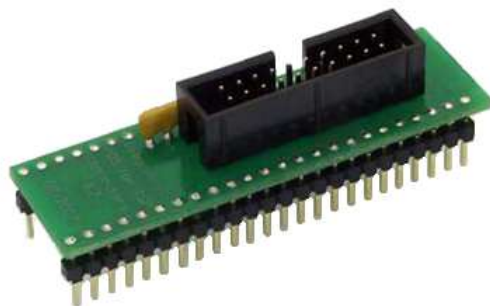
プログラマのセルフテストとキャリブレーション

M1883プログラマ使用中に異常を感じた場合にはセルフチェックを行なうことが出来ます。
セルフチェックには、M1883プログラマ購入時に付属しているセルフチェック用PODを使用します。

（1）セルフテスト1（プログラマの機能試験）
プログラマのメニューバーから プログラマ->セルフテスト を実行してください。
書き込みに必要なデバイス用電源回路、ドライバー、インターフェイス等をチェックします。

（2）セルフテスト2（ZIFソケットを含めた試験）
付属の48ピン“DiagnosticPOD1”（下図参照）をプログラマのZIFソケットに実装してください。
プログラマのメニューバーから プログラマ->セルフテスト プラス を実行してください。
書き込みに必要なデバイス用電源回路、ドライバー、インターフェイス等のチェックとZIFソケットに出力する信号をチェックします。

ZIFソケット検査用 “DiagnosticPOD1”

（3）セルフテスト3（ISPコネクタを含めた試験）
付属の48ピン“DiagnosticPOD2（for ISP）”（下図参照）をプログラマのZIFソケットに実装してください。この”DiagnosticPOD2“上の20ピンコネクタとプログラマ前面のISPコネクタを付属の20ピンフラットケーブルで接続してください。
プログラマのメニューバーから プログラマ->セルフテスト ISPコネクタ を実行してください。
書き込みに必要なデバイス用電源回路、ドライバー、インターファイス等のチェックとISPコネクタに出力する信号をチェックします。

ISPコネクタ検査用 “DiagnosticPOD2（for ISP）”

（4）キャリブレーション試験

プログラマ検査用の機能ですので、使用できません。
プログラマメーカーでの出荷試験、および校正時に使用する機能試験です。

M1883プログラマの基本仕様

ハード仕様

USB2．0ハイスピード対応、転送レート480Mbit/s
パラレル・ポート、転送レート1MB/s
FPGAベースのステート・マシーン
3個の高分解能D/A コンバータ（VCCP, VPP1、VPP2）
VCCP　電圧範囲　0～8v、最大電流1A
VPP1、VPP2　電圧範囲　0～26v、最大電流1A
セルフテスト機能付
電源入力部、パラレル・ポート接続部にESDプロテクション付
静電気対策リストバンド用のバナナジャック端子装備
アースGND接続用のバナナジャック端子装備

ソケット，ピンドライバー

300／600mil兼用　48ピンZIF(Zero Insertion Force)ソケット
ピンドライバー：48ピンの高機能ユニバーサルドライバー
48ピンのすべてにVCCP / VPP1 / VPP2電源を接続可能
48ピンのすべてにGND接続可能
FPGAベースのTTLドライバーで48ピンのすべてにH、L、CLK出力可能
アナログ・ドライバーとして使用する場合：出力範囲1．8v～26v
電流制限、過電流検出時シャットダウン、電圧異常時シャットダウン機能内蔵
ZIFソケットの各ピンにESDプロテクション付
（IEC1000-4-2、気中放電：15kV以上、接触放電：8kV以上）
デバイス書込み動作前に使用するピンドライバーをチェック

ISP コネクタ

20ピンフラットケーブル用オス・タイプ（後挿入防止付）
6ピン分のTTLドライバー（H、L、CLK、プルアップ、プルダウン）2v～5vをサポート
VCC電源：1回路　電圧範囲　2v～7v　最大電流100mA
シンク／ソース可能、電圧検出機能あり
VPP電源：1回路　電圧範囲　2v～25v　最大電流50mA
ターゲット・システム用電源：1回路　電圧範囲　2v～6v　最大電流250mA
ISPコネクタの各ピンにESDプロテクション付
（IEC1000-4-2、気中放電：15kV以上、接触放電：8kV以上）
動作結果を外部に出力する“GOOD”、“ERROR”信号をサポート
外部からプログラマを動作させる、スタート“YES!”用入力信号に対応

各種パッケージをサポート

DIPタイプのデバイスはプログラマ上の48ピンZIPソケットで対応
パケージ及びピン配置変換用に準備したアダプタは
DIP, SDIP, PLCC, JLCC, SOIC, SOP, PSOP, SSOP, TSOP, TSOPII, TSSOP, QFP,
PQFP, TQFP, VQFP, QFN (MLF), SON, BGA, EBGA, FBGA, VFBGA, UBGA, FTBGA,
LAP, CSP, SCSP etc

書き込み時間

Device	Size [bits]	Operation	Time
H26M11002AAR (eMMC NAND Flash)	3C780000hx8 (8 Giga)	programming *1	480 sec.
K8P6415UQB (parallel NOR Flash)	400100hx16 bit (64 Mega)	programming and verify	13 sec.
K9F1G08U0M (parallel NAND Flash)	8400000Hx8 (1 Giga)	programming and verify	122.7 sec
QB25F640S33 (serial Flash)	800200Hx8 (64 Mega)	programming and verify	30.7 sec
AT89C51RD2 (microcontroller)	10000Hx8	programming and verify	14.4 sec
PIC32MX360F512L (microcontroller)	80000Hx8	programming and verify	8.9 sec

Conditions: *P4, 2.4GHz, 512MB RAM, USB2.0, Windows XP*

＊1：カードリーダと同じ方法でアクセスしています。書込み後のベリファイはデバイス内のコントローラを使用しています。

ソフトウェアの仕様

デバイス用アルゴリズムはメーカー推奨のアルゴリズムを使用しています。ソフトウェアのアップデートは常時行なっています。
新しいデバイスが開発された時やデバイスの書き込みに変更があった場合には、最新のアルゴリズムを使用して書き込みができるように、アルゴリズムの更新と作成を随時行なっています。
Ｍ１８８３コントロールソフトをアップデートすることにより、アップデート時点の最新アルゴリズムに更新することが出来ます。
コントロールソフトの最新版へのアップデートは、代理店や営業所に問い合わせください。

デバイス関連操作

デバイス選択

デバイスTYPE、デバイスメーカー名、デバイス名の一部分を入力して一覧表示から選択、
EP－ROM、FLASH－ROMのAUTO＿IDを使用してデバイスを自動選択、
Blank、Read、Verify、Program、Erase動作、
コンフィグレーションとセキュリティー・ビット書き込み、
チェックサム、
JAM標準テストとSTAPLプログラミング言語及びJEDEC標準“JESD－71”のインタープリタ（解釈プログラム）を実装
バイナリー形式のSVFファイルを圧縮したVMEファイルのインタープリタ（解釈プログラム）を実装

デバイスチェック機能

デバイスの誤挿入チェック
ZIFソケットーデバイス間のコンタクトチェック
デバイスIDチェック

その他の機能

マスプロダクションモード（大量生産）をサポート ——＞デバイス装着後すぐに自動開始
（Automatic YES）
自動でデバイスにロット番号を書き込むモード（シリアライズモード）
生産個数をカウント表示
一定の数量書き込み後に自動停止するカウントダウンモード

バッファメモリ操作

バッファデータ（デバイスに書き込むデータ）の表示、編集、検索、置き換え
バッファデータのイニシャライズ、コピー、移動、バイトスワップ
バッファデータのチェックサム計算

ファイルの読込、保存

データフォーマット形式は自動検出またはマニュアル設定

サポートしているデータフォーマット形式
- バイナリーフォーマット（no＿format）
- HEX形式：INTEL、Motorora、Tektronix、ASCII＿HEX、ASCII＿SPACE
- Ascii形式：AltetaPOF、ABEL用JEDEC（ver3．0A）、CUPL、PALASM、
- JAM（JEDEC STAPL形式）、JBC（JAM STAPL ByteCode）、TANGO_PLD
  STAPL（STAPLファイル）、JEDEC standerd JESD－71
- VME（ispVMEファイル VME2．0／VME3．0）
- SVF（Serial Vector Format revision E）
- STP（Actel STAPL file）

一般仕様

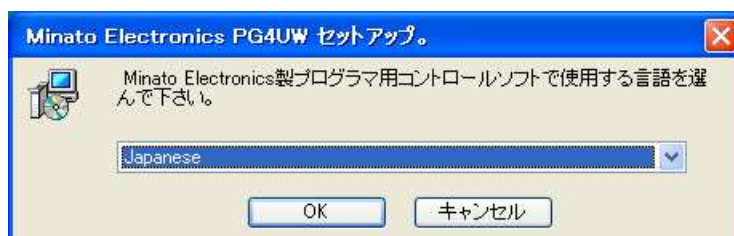
供給電源	AC100－240V 50－60Hz
消費電力	20W(動作時)、2W非動作時
寸　法	195(奥行き)x140(幅)x55(高さ)mm
質　量	0．9kg(ケーブル・ソケットアダプタ類を除く)
動作温度	5℃－40℃
動作湿度	20%－80%(結露がないこと)

インストール

梱包箱にはM1883プログラマ用コントロールソフトと取扱説明書を記録したCD－ROMが入っています。
プログラマとパソコンを接続する前にソフトウェアのインストールをしてください。

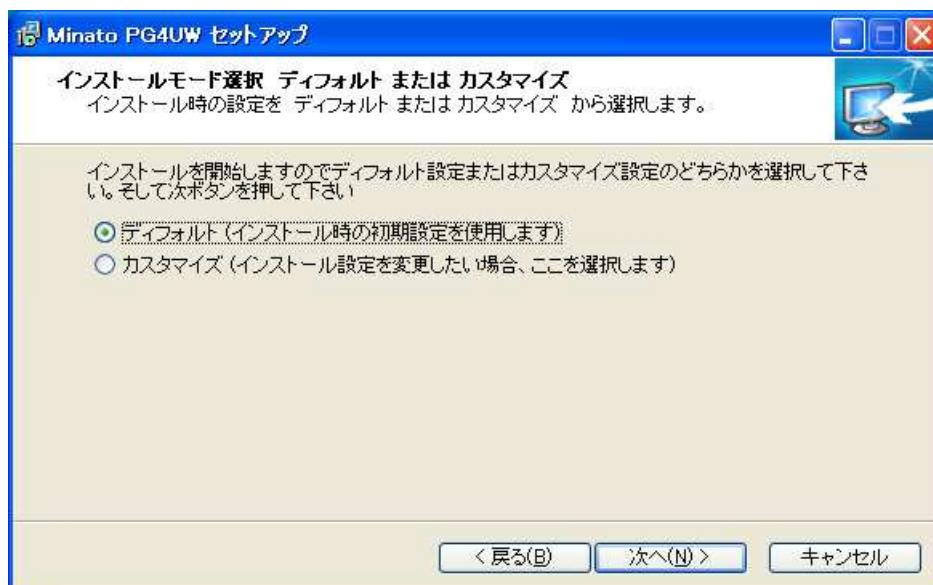
ソフトウェアをセットアップする

（プログラマとパソコンはまだ接続してはいけません。）
ソフトウェアをセットアップするパソコンのCDドライブにM1883プログラマに付属しているCDを入れて、"pg4uwarm. exe" を実行してください）
M1883プログラマ用コントロールソフトを動作させる前に、必要な全てのインストールを行ないます。

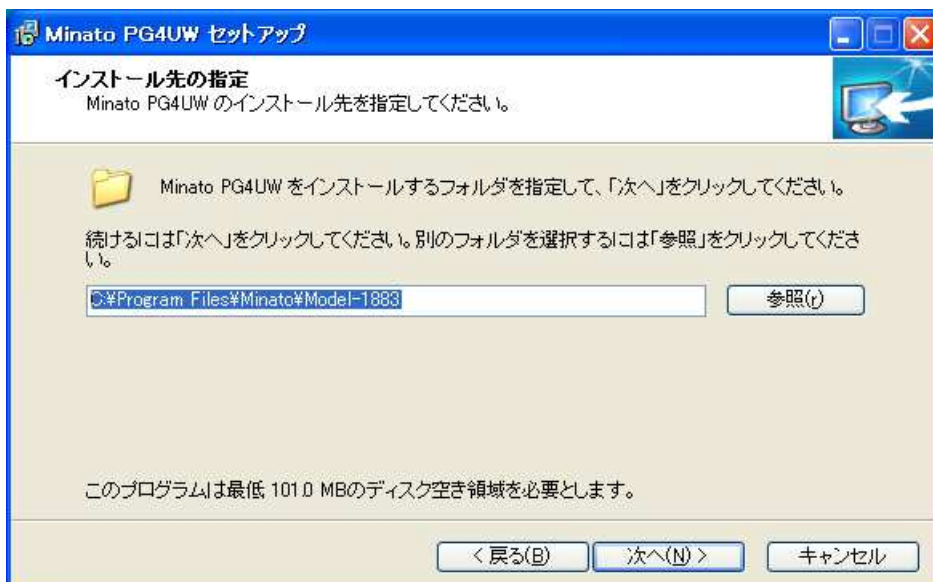
Setup　1

使用する言語を設定し、 OK をクリックしてください。

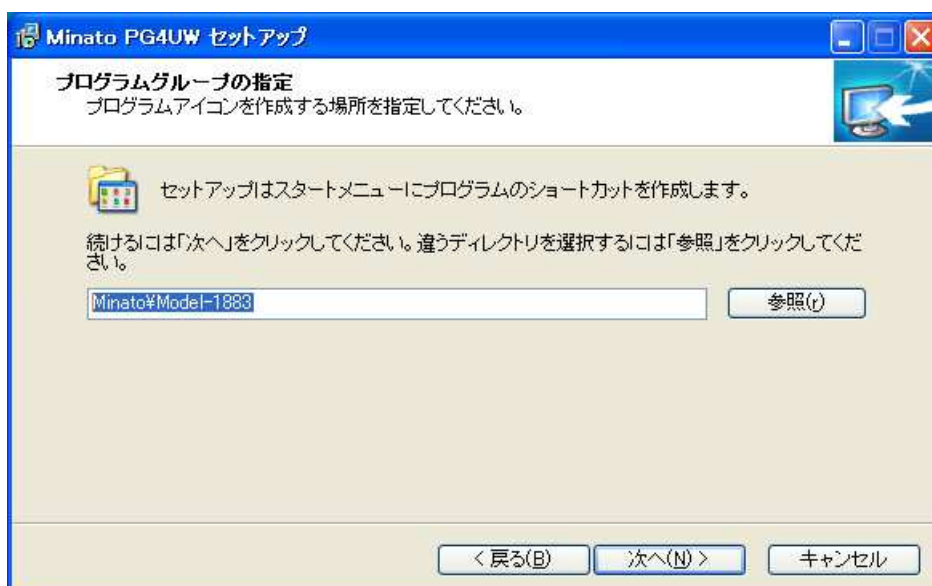
Setup　2

セットアップ開始の案内が表示されます。 次へ（N） をクリックしてください。

Setup 3

インストール時の設定をディフォルト設定にするかカスタマイズするか選択します。
カスタマイズ設定では、コントロールソフトをインストールするフォルダー指定と
ショートカットアイコンの登録指定が追加されます。

ディフォルト設定を選択し 次へ（N） をクリックしてください。

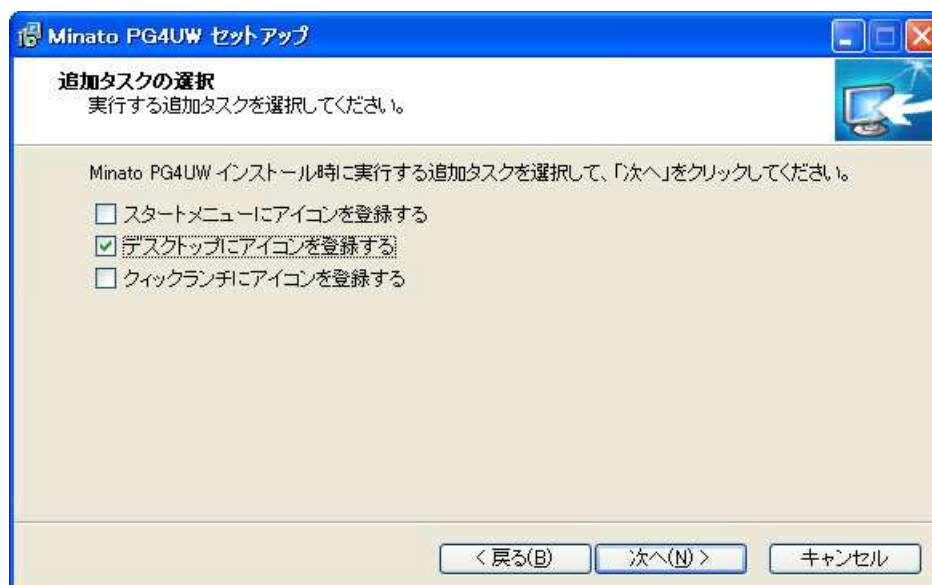
Setup 4 （ディフォルト設定を選択した場合は表示されません）

コントロールソフトをインストールするパソコン上のフォルダーが表示されます。
必要に応じて、インストールするフォルダーを変更してください。
確認後または変更後 次へ（N） をクリックしてください。

Setup　5 （ディフォルト設定を選択した場合は表示されません）

プログラマ起動時に使用するスタートメニューグループを指定します。

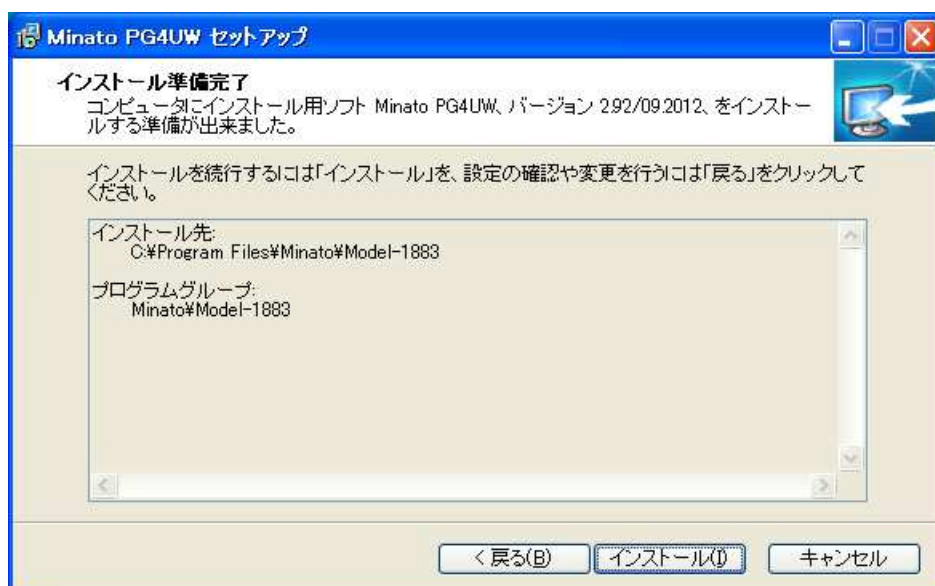
・既にスタートメニューに "Minato"が登録されていて、このグループ内に "Model－1883"名で登録する場合は "Minato￥Model－1883" で指定します。

・新規にスタートメニューに "新登録名xxx"で登録する場合は "新登録名xxx" と入力して指定します。

確認後または指定後 ［次へ（N）］ をクリックしてください。

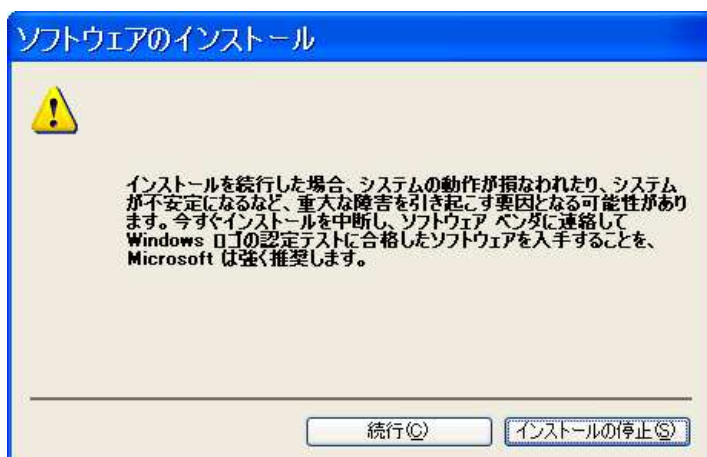
Setup　6 （ディフォルト設定を選択した場合は表示されません）

ショートカットの登録場所を指定します。

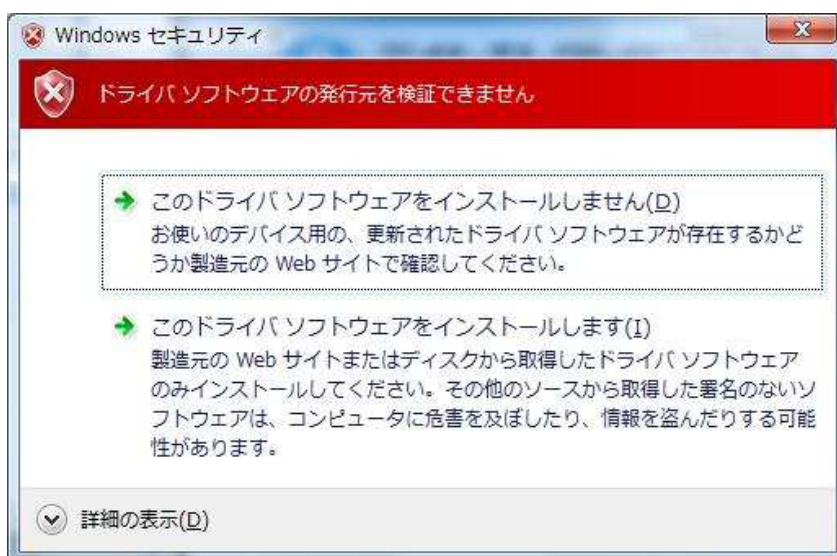
登録したい場所にチェックマークを付け ［次へ（N）］ をクリックしてください。

Setup 7

セットアップの準備が整いましたので、再確認の表示がでます。
インストールホルダを確認後 インストール(I) をクリックします。

Setup 8

インストール開始後上記の進行状態表示バーが表示されます。
セットアップ時間は2，3分です。

Setup 9

初めてインストールする場合、注意メッセージが表示されます。
続行（C） をクリックしてください

ビスタ版のパソコンの場合

上記表示が出たら “このドライバソフトウェアをインストールします”
をクリックします。

Setup　10

上記の表示がでれば、インストール完了です。

・プログラマとパソコンの接続に“USB”を使用する場合は再起動の必要はありません。

・パラレルI／Fを使用する場合はパソコンを再起動して下さい。

ハードウェアをセットアップする

M1883コントロールソフトを起動する前にM1883プログラマとパソコンをUSBケーブルで接続します。Windowsシステムは新しいハードウェアを検出しますので、USBドライバーのインストール方法を指定します。
プログラマが正常に検出され、M1883プログラマのインストールが開始されます。

Setup 1　USB（LPT）ケーブルをプログラマに接続します。

Setup 2　USB（LPT）ケーブルをパソコンに接続します。
（推奨はUSB2．0ハイスピード）

Setup 3　電源ケーブルをコンセントに接続します。
（延長ケーブルは使用しないでください）

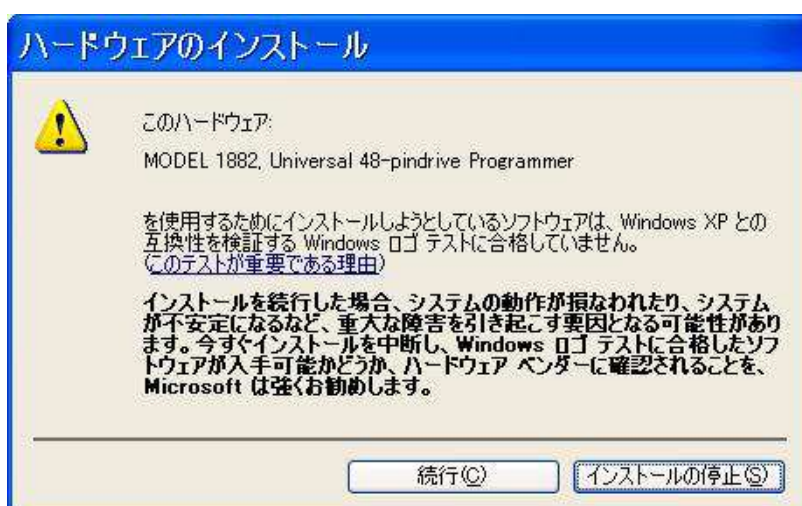
Setup 4　M1883プログラマの電源SWを入れてください。
全てのLEDが一時的に点灯し、プログラマに異常がない場合POWER_LEDが薄く点灯し、他のLEDは消灯します。
LPT接続の場合はこれで準備完了です。
USB接続の場合は次のステップに進みます。

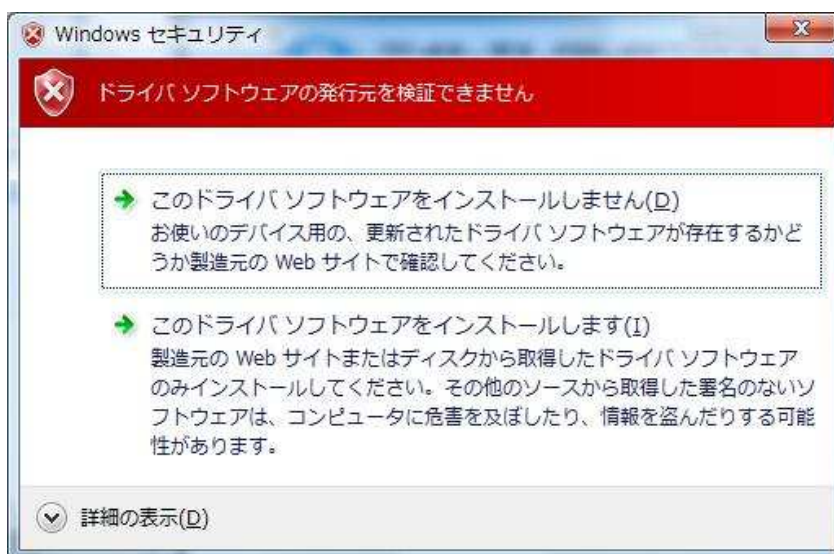
Setup 5　Winowsシステムは“新しいハードウェアの検出ウィザード”をスタートします。
Windows7の場合
“新しいデバイスソフトウェア”が自動インストールされます。

WindowsXP　SP2の場合

ここでは“自動的にインストールする”にチェックマークをつけて
次へ（N） をクリックします。

プログラマＭ１８８３が検出され、対応ＵＳＢドライバーがインストールされます。

Setup　6

続行（C） をクリックします。

Windows　Vista版では

上記表示が出たら"このドライバソフトウェアをインストールします"をクリックします。

Setup 7

完了 をクリックします。
これでハードウェアのセットアップが完了です。

Setup 8　　USBポートを変更した場合

セットアップを行なったUSBポート以外を使用した場合、新しいハードウェアの
セットアップWizardが起動し、使用するUSBポート別にドライバーが設定されます。
Step5. からの操作が必要になります。

以上でソフトウェアとハードウェアのセットアップは完了です。

第2章
M1883コントロールソフト　操作マニュアル

M1883 コントロールソフト

M1883コントロールソフトの実行

デスクトップ画面上のアイコンを をクリックしてください。

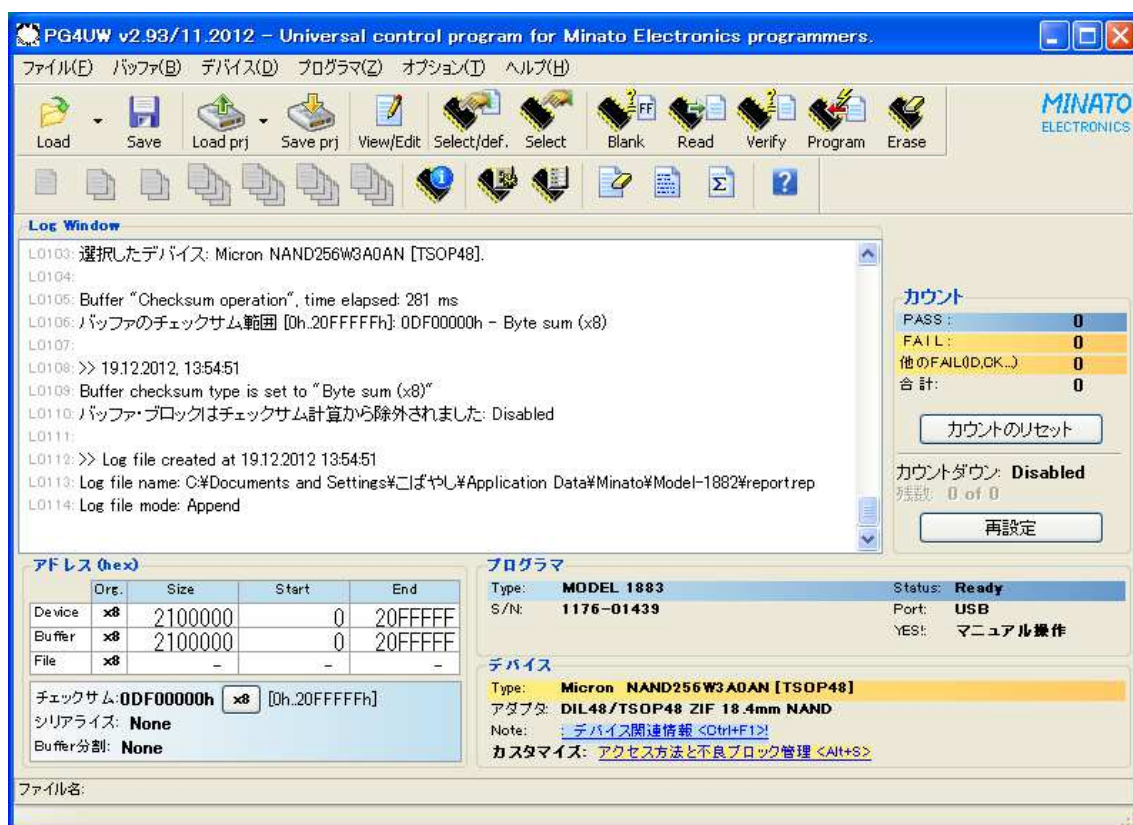
プログラマ検出画面にて、“MODEL1883” と接続するインターフェイス “USB” を選択して
接続(C) をクリックします。

検出後、メイン画面を表示し、ユーザーからの指示を待ちます。
コントロールソフトがプログラマを検出できなかった場合、画面上にエラー番号と考えられるエラー内容を表示します(プログラマの接続が外れている、間違った接続をしている、電源がOFFしている、プリンターポート設定ミス. .)
プログラマと接続したパソコンを再点検し、問題を取り除いてから、再実行してください。
エラーが再度出る場合は、コントロールソフトはデモ・モードで再開されますので、プログラマへのアクセスは出来ません。
もし、エラーの原因が見つからないときは営業所または代理店に連絡してください。
コントロールソフトはデバイスの書き込み開始時にもプログラマとの接続をチェックしています。

日本語表示に変更する

最初にコントロールソフトを起動したときは、英語表示になっています。
次の操作で日本語に変更出来ます。

TOPメニューの**オプション** —> **オプション設定** 内の
・言　語 ー>選択する言語 ー>“Japanese”
・言　語 ー>選択するヘルプファイル ー>“Japanese”
を選択してください。
コントロールソフトを一度終了してから、コントロールソフトを再起動してください。
日本語表示に変更されます。

メイン画面の説明

Toolbars（ツールバー）

メインメニューの下に配置したアイコンの集まりで、使用頻度の高いコマンドをアイコン化してあります。
このツールバーはメニュー上の<u>オプション -> ツールバー</u>で表示のON／OFF指定が出来ます。

Log・Window（ログ・ウィンドウ）

コントロールソフトが実行するプロセス、結果、情報を表示するウィンドウです。
Log・Windowに表示される情報は、表示と同時にファイルに記録することが出来ます。
Log情報はメニュー上の<u>オプション -> オプション設定 -> ログ・ファイル</u>で保存するファイルを指定出来ます。

アドレス（アドレス・ウィンドウ）

選択したデバイスの容量、開始アドレス、終了アドレス情報を表示します。
バッファメモリの容量、開始アドレス、終了アドレスを表示します。
読み込んだデータファイルの容量、開始アドレス、終了アドレスを表示します。
一部のデバイスでは、開始アドレス、終了アドレスが変更出来ます。
変更はメニュー上の<u>デバイス -> デバイスオプション -> 動作オプション</u>で行ないます。

アドレス・ウィンドウにはデバイスのチェックサム（バッファメモリのチェクサム計算値）、シリアライズ（連続番号書き込み）モード、Buffer分割（データ分割書き込み）を表示します。

これらの詳細は次のメニューを参照してください。
・チェックサム　　　：バッファ -> CheckSUM
・シリアライズ　　　：デバイス -> デバイスオプション -> シリアライズ
・Buffer分割　　　　：デバイス -> デバイスオプション -> 動作オプション

プログラマ（プログラマ・ウィンドウ）

使用しているプログラマ機種名および設定情報を表示します。
・Type　　　　　　　：プログラマの機種名
・Status　　　　　　：パソコンとプログラマの接続状態（Status）
　　Ready　　　　　　：プログラマ使用可能状態を示す
　　Not found　　　　：プログラマが検出できないことを示す
　　Demo モード　　　：OFF・ラインモード中を示す
・S/N　　　　　　　　：接続したプログラマの製造番号
・Port　　　　　　　　：プログラマと接続しているインターフェイス名
・YES！　　　　　　　：デバイスを書き込む時のスタート方法
　　マニュアル操作　：プログラマ上の（YES！ボタン）を押す、またはデバイス動作完了時に表示される“リピート確認”を使用する。
　　自　動　　　　　：デバイスの抜き差しをプログラマが検出し、自動で書き込みをスタートするモード

デバイス（デバイス・ウィンドウ）

選択しているデバイス情報を表示します。
・Type　　　　　　　：デバイス名、メーカー名を表示
・アダプタ　　　　　：デバイスに適合する変換アダプタを表示
・Note　　　　　　　：デバイスの詳細情報を表示するためのショートカット
・カスタマイズ　　　：デバイスのオプション機能を設定するためのショートカット

カウント（書き込み数量・ウィンドウ）

作業しているデバイスの数量を表示します。
・PASS　　　　　　　：正常終了したデバイスの数量
・FAIL　　　　　　　：データFAILになったデバイスの数量
・他のFAIL（ID, CK…）：IDチェックやピンチェックでFAILになったデバイスの数量
・合　計　　　　　　：PASS数 と FAIL数の合計

・カウントダウン　　：Enable又は、Disableを表示
　　カウントダウンモード時はここに作業中の残数を表示します。

カウントおよびカウントダウンの設定は
メニューの デバイス -> デバイスオプション -> カウント＆カウントダウン
またはカウント・ウィンドウ上でマウスを右クリックして、POPアップアイコンから変更出来ます。

ショートーカット・キーのリスト

キー	機能	説明
<F1>	ヘルプ	:操作マニュアル表示
<F2>	Save	:バッファメモリ内容をファイルに保存
<F3>	Load	:ファイルをバッファメモリに読み込む
<F4>	View/Edit	:バッファメモリの表示／編集
<F5>	Select／def.	:過去に使用していたデバイス・リストからデバイスを選択
<Alt+F5 >	Select	:デバイス選択
<F6>	Blank	:ブランクチェック
<F7>	Read	:デバイスの内容をバッファメモリに読み込む
<F8>	Verify	:デバイスとバッファメモリ内容を比較
<F9>	Program	:デバイスにバッファメモリ内容を書き込む
<F10>	Erase	:デバイスの内容を消去する
<Alt+Q>	終了	:設定を保存せずにコントロールソフトを終了
<Alt+X>	保存して終了	:設定を保存してコントロールソフトを終了
<Ctrl+F1>	デバイス情報	:現在のデバイス情報を表示する
<Ctrl+F2>	Erase buffer	:バッファメモリをイニシャライズする
<Ctrl+Shift+F2>	Fill random data	:ランダム値でバッファメモリを埋める

ファイル（メインメニュー・コマンド）

ファイルを操作するコマンドを集めたメニューです。
デバイスに書き込むデータファイルの読込／保存、プログラマの設定をまとめて管理するプロジェクトファイルの読込／保存、プログラム終了コマンドを集めたものです。

ファイル－>読込

デバイスに書き込むためのデータファイルをパソコン上のバッファメモリに読み込みます。
読み込み可能なフォーマットは
Binary、MOTOROLA、MOS_Technology、Tektronix、Intel_Hex、
ASCII_Hex、ASCII_Space、Straight_Hex,JEDEC、POFが使用出来ます。
コントロールソフトは最後に使用したデータファイルの拡張子を順次記録しています。

特別フォーマット：
IntelHEXx16 ：デバイス TMS320Fxxxx 専用に作られた IntelHEX タイプの 16bit 版です。
MotorolaHEXx16：デバイス TMS320Fxxxx 専用に作られた MotorolaHEX タイプの 16bit 版です。

ファイルフォーマットの説明

ASCII HEX フォーマット

1 バイトデータは、2 個の 16 進数で表され、スペースによって区切られています。
アドレスは、$Annnn の形式で表現されます。nnnn 部分は 4 個の 16 進数アドレスです。後ろにコンマが必要です。明白なアドレスがデータ列に含まれていない場合、ファイルデータは連続データとして扱われます。
最初のデータの前にアドレス部分が無い場合は、このデータはアドレスの 0000 番地として扱われます。このデータファイルの先頭は、STX コード(0×02)で始まり、データ部分の終了は ETX(0×03)で終わります。
注意: チェックサム部分は$Sxxxx の形式で表現されます。xxxx 部分は 4 個の 16 進数チェックサムです。チェックサムはこのファイルの最後の部分になります。

ASCII HEX ファイル例：
この例では"Hello, World"の文字列をアドレスの 1000 番地から入れています。
STX $A1000,
48 65 6C 6C 6F 2C 20 57 6F 72 6C 64 0A ETX
$S0452,

ASCII SPACE フォーマット

非常に単純なフォーマットで、ASCII HEX フォーマットのチェックサムとスタートコード(STX)とエンドコード(ETX)を取り払ったフォーマットです。
1 バイトデータは、2 個の 16 進数で表され、スペースによって区切られています。
アドレスは、4-8 個の 16 進数で表示されています。アドレスとデータはスペースで区切ります。
16 進数が 4 個以上連続しているとアドレスとして扱われます。

ASCII SPACE ファイル例：
この例では"Hello, World"の文字列をアドレスの 1000 番地から入れています。
0001000 48 65 6C 6C 6F 2C 20 57 6F 72 6C 64 0A

Straight HEX フォーマット

非常に単純なフォーマットで、ASCII HEX フォーマットのアドレスとチェックサムとスタートコード(STX)とエンドコード(ETX)を取り払ったフォーマットです。
1 バイトデータは、2 個の 16 進数で表され、スペースによって区切られています。

Straight HEX ファイル例:
この例では"Hello, World"の文字列を入れています。
48 65 6C 6C 6F 2C 20 57 6F 72 6C 64 0A

Samsung HEX フォーマット

Samsung HEX フォーマットは、Intel HEX フォーマットを少し修正したものです。ファイルの形式及び構成等は Intel HEX フォーマットと同等です。

File format

・ファイルフォーマットの自動認識: チェック有り時
ファイル読込時にデータフォーマットを自動識別します。
・ファイルフォーマットの自動認識: チェック無し時
"選択ファイルフォーマット" のリストボックス内から使用するフォーマットを選択します。

フォーマット自動識別か手動設定かの指定は、
メニュー **オプション -> オプション設定 -> Loadファイル オプション**の中にある
読込時のファイルフォーマット 項目で指定します

注意: ASCII HEX 形式のファイルが自動認識できなかった場合、この ASCII HEX 形式ファイルはバイナリーファイルとして読み込まれます。ASCII HEX 形式のファイルを読み込む場合はフォーマットの手動設定を推奨します。

追加オペレーション

追加オペレーションでは次の設定が出来ます。
・Byte Swap:チェック有り時
ファイル読み込み中に 16 ビットデータを 8 ビット(1Byte)単位でスワップ(入れ替え)を行ないます。この機能はビッグ・エンディアンで記録されているモトローラ形式のファイルをリトル・エンディアンでバッファメモリにリードするときに便利です。
標準の読込ファイルではリトル・エンディアンのバイト形式を使用します。
・読込み前にバッファデータ消去:チェック有り時
ファイルを読み込む前にバッファメモリ内容を指定したデータでクリアします。
このディフォルト値は、
メニュー **オプション -> オプション設定** の中にある Load ファイル オプションで変更出来ます。
・Blank スペアエリアを追加する:(NAND デバイス専用):チェック有り時
ファイル読み込時に NAND デバイス特有の spare エリアを追加したい場合にこのオプションを設定します。追加される spare エリアは設定する NAND デバイスに依存します。

バッファオフセット

バッファオフセットでファイルから読込んだデータをバッファに入れる時のオフセットアドレスを指定出来ます。
標準設定はいつも"None"になります。ユーザーが使用する時にオフセット値を設定して下さい。
次の設定が出来ます。

・**None**:読込んだファイルはオフセット無しでバッファメモリに入ります。

・**ポジティブオフセット**:読込んだファイルアドレスにオフセット値を加算したバッファアドレスにデータをストアします。Binary、Hexの両方のフォーマットで使用できます。
バッファアドレス ＝ フォーマットアドレス ＋ オフセットアドレス

・**ネガティブオフセット**:読込んだファイルアドレスからオフセット値を引き算したバッファアドレスにデータをストアします。
バッファアドレス ＝ フォーマットアドレス ー ネガティブオフセット
ネガティブオフセットはHexフォーマットのみの対応になります。

・**Automatic ネガティブオフセット**: この設定では読込むファイルから自動的にオフセット値を検出して使用します。
バッファアドレス ＝ フォーマットアドレス ー ネガティブオフセット
ネガティブオフセットはHexフォーマットのみの対応になります。

ネガティブオフセット使用時の注意

・ネガティブオフセットは HEX ファイルフォーマット時に有効ですが、バイナリーファイルでは使用できません。
・ネガティブオフセットはファイルアドレスからオフセット値を単純に引き算しています。
そのため計算値がマイナスになった場合には、データをバッファに読込めません。
ネガティブオフセット値は使用するファイルにあった適正値を設定して下さい。
・Automatic ネガティブオフセットは特別な場合以外は使用しないで下さい。
このモードは読込むデータファイルの構成が不明な場合や、壊れたファイルの一部を読むなどのファイル解析的なモードです。
・ Automatic ネガティブオフセットはいくつかの特別なデバイスでは使用出来ません。データを LOAD するデバイスの Block 番号が指定されている場合は使用出来ません。
例:Microchip PIC micro Device。
このデバイスを使用する場合オフセットのマニュアル設定を使用して下さい。

ネガティブオフセットの使用例

-ファイルフォーマットは MOTOROLA-S
-ファイル上で使用するデータのアドレス FFFF0h 以降のデータ
-プログラマのオフセットアドレスに FFFF0h を設定します。

ファイルの読込みを実行すると、アドレス FFFF0h より前のデータは読み飛ばし、FFFF0h 以降のデータがバッファメモリの 0 番地から順にストアされます。

ファイルフォーマット番号とエラー番号

読み込みエラーが発生した場合、下記のエラー番号をLog・Windowに表示します。

"Warninig errorr # XXY in line RRR"

XX　はフォーマット番号

Y　 は下記のエラーコード

RRR はエラーが発生したファイルのライン番号(10進数)

File format codes:

#00y – binary
#10y – ASCII Space
#20y – Tektronix
#30y – Extended tektronix
#40y – Motorola
#50y – MOS Technology
#60y – Intel HEX

Load file error codes:

#xx1 – bad first character – header
#xx2 – bad character in current line
#xx3 – bad CRC
#xx4 – bad read address
#xx5 – bad length of current line
#xx6 – too big negative offset
#xx7 – address is out of buffer range
#xx8 – bad type of selected file format
#xx9 – the file wasn't loaded all

ファイル–>保存

マスターデバイスからリードしたデータはパソコン上に用意したバッファメモリに一時的に保持しています。保存コマンドはこのバッファメモリデータをパソコン上のファイルに記録保存する機能です。
ファイルに記録するときに、データのファイルフォーマットと記録するデータの開始アドレスや終了アドレスを指定出来ます。
使用できるフォーマットは

Binary、MOTOROLA、MOS_Technology、Tektronix、Intel_Hex、
ASCII_Hex、ASCII_Space、Straight_Hex,JEDEC、POFが使用出来ます。

保存時のオプション：保存するバッファメモリの開始アドレスと終了アドレスを指定することが出来ます。

Buffer Start = **保存するバッファメモリの開始アドレスを入力します**

Buffer End = **保存するバッファメモリの終了アドレスを入力します**

追加オペレーション：

・**Byte Swap**： チェック有りの場合（Byte Swap ON）
16ビットデータを8ビット（1Byte）単位でスワップ（入れ替え）してファイルに保存します。
標準のファイル保存（Byte Swap OFF）ではリトル・エンディアンのバイト形式を使用します。
パソコンの＜F3＞keyでこのメニューを呼び出す事が出来ます。

ファイル–>プロジェクトの読込

プログラマで使用するプロジェクトファイルを読込む機能です。
プロジェクトファイルにはデバイスの設定情報とデバイスに書込む為のデータ、及び使用するインターフェイス情報等が記録されています。

プロジェクトの読込 ダイアログには、使用するプロジェクトファイルを選択する Window と、選択したプロジェクトファイルの概要を表示する Window があります。

使用するプロジェクトファイルを選択する Window で各プロジェクトファイルをマウスでクリックするとその内容が**プロジェクトの概要**ウィンドウに表示されます。
ここに表示されるプロジェクト情報は

・プロジェクト内で最初に指定されたデバイス名、メーカー名
・プロジェクトファイルを記録した日時
・ユーザーが追記したコメント

Note：Serialization を使用するプロジェクトについて

Serialization は次の手順に従って、プロジェクトファイルからリードされます。

1. プロジェクトに記述した Serialization 設定が受け付けられます。
2. Additional Serialization ファイルを検索します。ファイルが検出された場合、追加ファイルとしてリードされます。

Additional Serialization ファイルは特別なプロジェクトファイルとして関連付けされます。
そしてこのファイル設定が受け付けられると、それまでの Serialization 設定は無効になります。
Additional Serialization ファイル名はプロジェクトファイル名の拡張子が ″.sn″に変更されたファイル名になります。
Additional Serialization ファイルはプログラマ制御ソフトと同一のディレクトリ内の″serialization¥″に配置されます。

例： プロジェクトファイル名： my_work.prj
プログラマのディレクトリ： c:¥Program Files¥Programmer¥
Additional Serialization ファイルは：
c:¥Program Files¥Programmer¥serialization¥my_work.prj.sn
Additional Serialization ファイルはデバイスのプログラム成功後に作成され更新されます。

ファイル –> プロジェクトを保存は現在の保存プロジェクトに関連するファイルが存在する場合は、Additional Serialization ファイルを削除します。

作業識別用ID入力について（Job ID）

プロテクトされたプロジェクトファイルをリードした時、Job ID コードを入力する画面が表示されます。2 個の入力部分があります。

・作業者 ID – プログラマの作業者用 ID として使用します。パラメータは 3 文字以上必要です。
プロテクトされたプロジェクトの **Job Report** を作成する時に使用します。

・ Job ID 入力 – 現在行っている作業を示す ID を入力します。
ディフォルトとして、読込んだプロジェクトファイル名に年月日と時間が追記された名称が記入されています。

Job ID はパスワードではありません。**作業者 ID** も **Job ID** も Job Report に記録する作業情報のために入力します。

ファイル–>プロジェクトを保存

プロジェクトファイルを保存する機能です。
プロジェクトファイルにはデバイス設定情報とプログラマのバッファデータが含まれています。
プロジェクトファイルに保存されたデータはメニューの **プロジェクトの読込み** で、いつでも使用出来ます。

プロジェクトの概要(選択したファイルに記載されている内容)

ダイアログ内に選択したプロジェクトファイルの内容を表示しています。
このBOXは表示専用ですので、内容等の変更は出来ません。

プロジェクトの概要(保存されている内容)

上半分はプログラマで現在選択されているデバイス情報を表示しています。
選択されているデバイス名、日付、プログラマのバージョン．．等を表示しています。
内容等の変更は出来ません。

下半分はユーザーが記述可能な部分です。
通常はプロジェクトの内容や、コメント等を記録します。

Project protection settings

・プロジェクトファイルを暗号化する：

チェック有り時：プロジェクトファイルを暗号化して保存します。パスワード無しでプロジェクトファイルを読込む事を防止します。
Passwordボタン をクリックすると、暗号化に使用するパスワード入力画面が表示されます。
プロジェクト保存時に使用するパスワードを指定して下さい。

・プロジェクトファイルを読込後、プロテクトモードに設定する:

チェック有り時：プロジェクトファイルを読込後、プロテクトモードに設定します。
Passwordボタン をクリックすると、プロジェクト保存時に使用するプロテクトモード用パスワード入力画面が表示されます。
他のプロジェクトファイルを間違ってリードしてしまう事を防止します。
プロテクトモード状態でプロジェクトを保存することを、**Protected mode project** と言います。詳細は **オプション -> 操作プロテクトモード** を参照して下さい。
操作プロテクトモードが有効になった場合は、Log Window上の右上部分に **Protected mode** を表示します。

設定するパスワードについて：プロジェクトファイル暗号化用パスワードとプロテクトモード設定用パスワードには別々の異なるパスワードをセットして下さい。

・プログラム開始前に右記のプロジェクトファイル ID 番号を入力する:

チェック有り時：プロジェクトファイルを読込み後の最初のデバイス書込み前に、実行するプロジェクトファイル ID 番号入力が必要になります。
このチェックが設定されたプロジェクトファイルを読込み後、間違ったデータ書込みを防止するためのモードです。

ファイル–>最近使用したファイル

最近使用したファイル履歴の中から、ファイルを選択して再読込みを行います。
ユーザーが使用するデータファイルを読込み(LOAD)すると、このファイル名はファイル履歴に追加されます。
ここで LOAD したファイルは、前回 LOAD したファイルの前に追加されます。

ファイルの再読込みの方法

1. メニューから ファイル –> 最近使用したファイル を選択します。
2. 最近使用したファイル名が 9 件 表示されます。この中から使用するファイルを選択して下さい。

ファイル–>最近使用したプロジェクト

最近使用したプロジェクトファイル履歴の中から、使用するプロジェクトファイルを選択して、再読込みを行います。

ユーザーが使用するプロジェクトファイルを読込みすると、このプロジェクトファイル名はファイル履歴に追加されます。
ここで 読込んだプロジェクトファイルは、前回読込んだプロジェクトファイルの前に追加されます。

プロジェクトファイルの再読込みの方法

1. メニューから ファイル –> 最近使用したプロジェクト を選択します。
2. 最近使用したプロジェクトファイル名が 9 件 表示されます。この中から使用するプロジェクトファイルを選択して下さい。

ファイル–>プロジェクト・オプション

現在プログラマで使用しているプロジェクトファイルの情報を表示する機能です。
またプロジェクトファイルにコメントを追記することが出来ます。
プロジェクト・オプションは以下の表示をおこないます。

- ・デバイス名、メーカー名
- ・プロジェクトファイルを記録した日時
- ・プロジェクトを作成した時のプログラマのバージョン番号
- ・ユーザーが追記したコメント

デバイス名やメーカーが表示されている部分はプログラマが自動的に作成しています。
ユーザーはその下の部分にプロジェクトの内容等をコメントとして入力することが出来ます。

ファイル–>Job Report..作成

現在プログラマで作業している情報を集約してレポートファイルを作成する機能です。
Job Reportはプロジェクトファイルの読込みから開始して、次のプロジェクトファイルの読込時、またはプログラマの作業終了(PG4UWソフト終了)時に作成されますが、このコマンドを実行することにより 現時点のJob Reportが作成されます。

ファイル−>e. tableの読込（encryption　table）

DISKからバイナリーファイルのデータを読み込みます。
このデータを、暗号テーブル用に確保したメモリ部分に入れます。
一部のデバイス専用です。

ファイル−>e. tableを保存（encryption　table）

暗号テーブル用に確保したメモリ部分の内容を、バイナリーデータとしてDISKのファイルに書込みます。
一部のデバイス専用です。

ファイル−>終了

現在のデバイス設定情報をDISKに保存しないで、プログラムを終了します。

ファイル−>保存して終了

現在のデバイス設定情報をDISKに保存し、プログラムを終了します。

バッファ（メインメニュー・コマンド）

バッファメモリを操作するコマンドを集めたメニューです。
バッファメモリとはターゲットデバイスに書き込むためにパソコン上に確保したメモリです。
カーソル Key で編集場所に移動して、データを変更します。変更されたデータはカラー表示されます。

バッファ ->表示／編集

バッファメモリの内容を見る（表示モード）、データを編集（編集モード）するときに使用します。
カーソルKeyで編集場所に移動して、データを変更します。またショートカット＜F4＞keyでもこのモードを起動出来ます。

Editメニューで使用できる　EditファンクションKey説明

＜F1＞（ Help ）：ヘルプ（説明）を表示します。

＜F2＞（ Fill buffer block ）
：指定した範囲のバッファメモリを指定した複数データで書き換えます。
複数データとして1～16byteまで指定できます。

＜Ctrl＋F2＞（ Ersae buffer block ）
：指定した範囲のバッファメモリのデータを指定したデータで書き換えます。

＜Ctrl＋Shift＋F2＞（ Fill Random data ）
：指定した範囲のバッファメモリデータをランダムデータで書き換えます。

＜F3＞（ Copy buffer block ）
：バッファメモリの指定した範囲のデータを他の場所に複写（COPY）します。

＜Shift＋F2＞（ Save 2nd Buffer data ）
：2ndバッファメモリデータを指定したバイナリーファイルに保存します。
この機能は2ndバッファメモリを持っている一部のデバイス選択時に使用できます。
（例：Data_EEPROM を持っている Microchip 社の PICmicro device など）。

＜Shift＋F3＞（ Load 2nd Buffer data ）
：指定したバイナリーファイルを2ndバッファメモリデータに読込みます。
この機能は2ndバッファメモリを持っている一部のデバイス選択時に使用できます。

＜F4＞（ Move buffer block ）
：バッファメモリの指定した範囲のデータを他の場所に移動（ MOVE ）します。

＜F5＞（ Swap data ）
：指定した範囲のバッファメモリ上の偶数アドレスのデータと奇数アドレスのデータの順番を入れ替（スワップ）します。

＜F6＞（バッファ印刷）
：バッファメモリの選択された部分をプリンター又は、ファイルに出力します。

＜F7＞(文字列検索)

:バッファメモリ内の文字列を検索します (ASCIIコードまたは Hexコード)。

＜F8＞(データ置換)

:バッファメモリ内の文字列を検索し、一致した文字列を置換します。
(ASCIIコードまたは Hexコード)。

＜F9＞(表示アドレス変更)

:カーソルが置かれているアドレスを表示するBoxです。この部分にアドレスを入力すると、入力したアドレスのバッファ内容を表示します。

＜F10＞(バッファ表示／編集)

:表示 / 編集モードを切り換えます。

＜F11＞(8bit / 16bit)

:バッファ表示を 8bit / 16bit 表示に切り換えます。

＜F12＞(Check sum)

:チェックサムオプション表示

＜矢印keys＞ :カーソル移動
＜Home／End＞ :現在行の先頭/最後へジャンプ
＜PgUp／PgDn＞ :前/次ページへジャンプ
＜Ctrl＋PgUp／PgDn＞ :現在のページの先頭行/最終行へジャンプ
＜Ctrl＋Home／End＞ :設定デバイス対象エリアの開始/終了へジャンプ
＜Shift＋Home／End＞ :エディット対象エリアの開始 / 終了へジャンプ
＜Backspace＞ :カーソルを1つ左へバック

バッファ ->表示／編集 buffer for PLD

PLDデバイスを選択した場合のEditファンクションKey説明

＜Ctrl＋F2＞(Ersae buffer block)
:指定した範囲のバッファメモリのデータを指定したデータで書き換えます。

＜Ctrl＋Shift＋F2＞(Fill Random data)
:指定した範囲のバッファメモリデータをランダムデータで書き換えます。

＜F9＞(表示アドレス変更)
:カーソルが置かれているアドレスを表示するBoxです。この部分にアドレスを入力すると、入力したアドレスのバッファ内容を表示します。

＜F10＞(バッファ表示／編集)
:表示 / 編集モードを切り換えます。

＜F11＞(1bit / 8bit)
:バッファ表示を 1bit / 8bit 表示に切り換えます。

＜矢印keys＞	:カーソル移動
＜Home／End＞	:現在行の先頭/最後へジャンプ
＜PgUp／PgDn＞	:前/次ページへジャンプ
＜Ctrl＋PgUp／PgDn＞	:現在のページの先頭行/最終行へジャンプ
＜Ctrl＋Home／End＞	:設定デバイス対象エリアの開始/終了へジャンプ
＜Backspace＞	:カーソルを1つ左へバック

注意:PLDデータはLogicデバイス開発用のToolが作成するデータです。
データの変更は十分注意して行なってください。

バッファ ->Fill block<ブロック書換え>

指定した範囲のバッファメモリを指定した複数データで書き換えます。
指定できるデータはHexデータ(0x00～0xFF)、またはASCIIデータで最大16バイト

開始アドレス　:書き換えを開始するバッファアドレス
終了アドレス　:書き換えを終了するバッファアドレス
書換えデータ　－＞Hex　:書き換えするデータをHexで入力
　　　　　　　－＞Ascii　:書き換えするデータをASCIIで入力

・**アドレス履歴を記録する**:を選択すると、使用したアドレス値が保存されます。この履歴は選択したデバイス毎に 15 個まで保存されます。

Note:この履歴は他のバッファ操作と共通です。
″最後に設定した値を使用する″を選択すると、次回のバッファ操作時に前回使用した最後のアドレス設定値が使用できます。

バッファ ->Copy block<ブロックコピー>

バッファメモリの指定した範囲のデータを他の場所に複写(Copy)します。

開始アドレス　:Copy元の開始バッファアドレス
終了アドレス　:Copy元の終了バッファアドレス
移動先　:Copy先の先頭バッファアドレス

・**アドレス履歴を記録する**:を選択する機能は“Fill Block”と同一です。

バッファ ->Move block<ブロック移動>

バッファメモリの指定した範囲のデータを他の場所に移動(Move)します。

開始アドレス　:Move元の開始バッファアドレス
終了アドレス　:Move元の終了バッファアドレス
移動先　:Move先の先頭バッファアドレス

注:転送元のデータはデバイスのBlankデータで書き替わります。

・**アドレス履歴を記録する**:を選択する機能は“Fill Block”と同一です。

バッファ ->Swap data<ブロック内でデータスワップ>

指定した範囲のバッファメモリ上の偶数アドレスのデータと奇数アドレスのデータの順番を入れ替(スワップ)します。

開始アドレス　:スワップを開始するアドレスを指定します。開始アドレスには偶数アドレスを指定する必要があります。偶数以外の場はプログラマが低い偶数アドレスに修正します。
終了アドレス　:スワップを終了するアドレスを指定します。終了アドレスには奇数アドレスを指定する必要があります。奇数以外の場合はプログラマが高い奇数アドレスに修正します。
16bitワード. . .　:16bitのワードデータを8bitのバイト単位でスワップします。
32bitワード. . .　:32bitのロングデータを8bitのバイト単位でスワップします。
4bit ニブル. . .　:1バイト内の上下4bitデータを1バイト内で入れ替えます。
1byteミラービット　:1バイト内のbit列の並びを逆にします。

スワップ、ニブル例：

バッファアドレス	変更前データ	16bitワード2byteスワップ	32bitワード4byteスワップ	4bitニブル	ミラービット
0x0000	byte0	byte1	byte3	byte0n	byte0m
0x0001	byte1	byte0	byte2	byte1n	byte1m
0x0002	byte2	byte3	byte1	byte2n	byte2m
0x0003	byte3	byte2	byte0	byte3n	byte3m
0x0004	byte4	byte5	byte7	byte4n	byte4m
0x0005	byte5	byte4	byte6	byte5n	byte5m
0x0006	byte6	byte7	byte5	byte6n	byte6m
0x0007	byte7	byte6	byte4	byte7n	byte7m

・byte0,byte1,byte2…はバッファ0x0,0x1,0x2…番地のデータ変更前を表す。
・byte0n,byte1n,byte2n…はニブルスワップ後のデータで
　元のデータのbit順（bit7 bit6 bit5 bit4　bit3 bit2 bit1 bit0）が
　ニブル後のbit順（bit3 bit2 bit1 bit0　bit7 bit6 bit5 bit4）になったデータを表す。
・byte0m,byte1m,byte2m…はミラー処理後のデータで
　元のデータのbit順（bit7 bit6 bit5 bit4　bit3 bit2 bit1 bit0）が
　ミラー後のbit順（bit0 bit1 bit2 bit3　bit4 bit5 bit6 bit7）になったデータを表す。

・**アドレス履歴を記録する**：を選択する機能は"Fill Block"と同一です。

バッファ ->Erase block<ブロック消去>

指定した範囲のバッファメモリのデータを指定したデータで書き換えます。

開始アドレス　：書換えを開始するバッファアドレス
終了アドレス　：書換えを終了するバッファアドレス
データ　：書き換えるデータを指定します。
（ディフォルト値は現在選択しているデバイスのBlankデータです）

ショートカット＜Ctrl＋F2＞keyが使用出来ます。
・**アドレス履歴を記録する**：を選択する機能は"Fill Block"と同一です。

バッファ ->Fill block with random data<ランダムデータセット>

指定した範囲のバッファメモリデータをランダムデータで書き換えます。
（デバイスの評価用データを作成出来ます。）

開始アドレス　：書換えを開始するバッファアドレス
終了アドレス　：書換えを終了するバッファアドレス

ショートカット＜Shift＋Ctrl＋F2＞keyが使用出来ます
・**アドレス履歴を記録する**：を選択する機能は"Fill Block"と同一です。

バッファ ->Duplicate buffer

小容量デバイス用に作成したマスターデータを大容量デバイスに書き込むとき、バッファメモリ上にマスターデータと同一のCopyデータを複数個作成する機能です。

例：装置で使用していたデバイス（i27C256）が製造中止になり、大きな容量のデバイス（i27C512）を使用する場合があります。
装置上の最上位アドレスが不明な場合、大容量デバイスの前半と後半に同一のデータを入れておけば、安全に動作させる事が可能です。

バッファ ->表示／編集->バッファ印刷

バッファメモリの選択された部分をプリンター又は、ファイルに出力します。
プログラムは外部テキスト・エディターを使用します。
ディフォルトは**Notepad. exe**が設定されています。

開始アドレス　：バッファメモリの開始アドレスを入力
終了アドレス　：バッファメモリの終了アドレスを入力
外部エディター　：バッファ内容を表示、編集、保存、Printするために使用する外部エディターのパスと名前を指定します。
ディフォルトでは**Notepad. exe**が設定されています。
他のテキスト・エディターを指定することも出来ます。

バッファ ->表示／編集->文字列検索

バッファメモリ内の文字列を検索します。検索文字列入力ボックスに検索したいデータ（Hexデータ）または文字列（ASCIIデータ）を入力します。16文字まで検索出来ます。

検索方向：検索する方向を指定します。
下方向　：現在の位置又は、バッファメモリの先頭から最後方向へ検索します。
上方向　：現在の位置又は、バッファメモリの最後から先頭方向へ検索します。
開始場所：検索を開始する場所を指定します。
カーソルから：現在の位置から検索を開始します。
先頭から　：全体を検索します。

バッファ ->表示／編集->データ置換

バッファメモリ内の文字列を検索し、一致した文字列を置換します。検索文字列入力ボックスに検索したいデータを、置換文字列入力ボックスに置換したいデータ入力します。16文字まで置換出来ます。

検索方向：検索する方向を指定します。
下方向　：現在の位置又は、バッファメモリの先頭から最後方向へ検索します。
上方向　：現在の位置又は、バッファメモリの最後から先頭方向へ検索します。
開始場所　：検索を開始する場所を指定します。
カーソルから：現在の位置から検索を開始します。
先頭から　：全体を検索します。
オプション　：このオプションを設定した場合、置換の前に確認用のメッセージを表示します。

置換時の確認用ダイアログの説明：**置換開始**ボタンを押すと、検索を開始します。
一致した文字列が発見された時、確認用ダイアログが表示されます。
Yes　：置換し、次を検索します。
No　：置換せずに次を検索します。
全置換　：すべてを置換します。
検索中止　：検索を中止します。

バッファ ->CheckSUM

チェックサム値はプログラマで使用するユーザーデータが正しいかを確認するために使用する値です。

チェックサムオプション用ダイアログ画面について

チェックサムオプション用ダイアログ画面には2個の設定タブがあります。

1. ダイレクトサム計算タブ
   アドレス範囲を指定してサム計算したい場合や、計算したサム値を特定のバッファに挿入する場合に使用します。
2. メイン画面のチェックサムタブ
   現在使用中のバッファデータのチェックサム値を自動的にメイン画面のアドレス(Hex)表示と Log Window に表示します。
   また一部のデータをチェック計算から除外する場合、ここで指定します。

ダイレクトサム計算タブ

アドレス範囲を指定してサム計算したい場合や、計算したサム値を特定のバッファに挿入する場合に使用します。

・チェックサム計算用のアドレス範囲指定部:

開始アドレス／終了アドレス:
チェックサムを計算するアドレス範囲を入力します。アドレスは Byte アドレスで入力して下さい。
このアドレス入力はチェック Box が "Enable" の時だけ有効になります。

・チェックサムから除外するバッファ指定部:

一部のデータをチェックサム計算から除外設定する場合に使用します。
この機能は、シリアライズ機能で使用します。
シリアライズ機能はデバイスに連続番号を書く機能ですが、それと同時にバッファデータの一部を変更しています。
サム計算からこの連続番号に相当するデータを除外しておけばシリアライズ機能に関係なく、一定のチェックサム値にすることが出来ます。

・チェックサム Result 表示部:

色々な種類のチェックサム値を表示する部分です。詳細は下記の部分を参照して下さい。
各サム値以外に Neg(反転)、Suppl(補数)を表示しています。

・挿入するチェックサムの設定部:

チェックサム挿入:
計算の実行と挿入がクリックされたときバッファメモリに書込むチェックサムの種類を選択する項目です。

挿入するアドレス:
計算の実行と挿入がクリックされたときに、選択されたチェックサムの結果を書込むバッファメモリのアドレスを指定します。
チェックサムを挿入するアドレスは**開始アドレス**から**終了アドレス**の範囲外を指定します。
具体的には、チェックサム計算する範囲をROMよりも小さくし、この範囲外にチェックサムを挿入します。

Size :
この項目は選択されたチェックサム結果が書込まれるバッファのサイズを設定するために使用されます。
サイズ指定は Byte (8-bit), Word (16-bit) または DWORD (32-bit)から指定します。
指定したサイズが選択したチェックサムよりも桁数が少ない場合は、下位バイトがバッファに書き込まれます。

・WORD／DWORD size 選択時
チェックサム値の Low バイトが挿入するアドレスで指定されたアドレスに書込まれ、そして、High バイトが次のアドレスに書込まれます。

・計算実行ボタン:
　チェックサムを計算します。バッファメモリへの書込みは行われません。
・計算実行　&　挿入ボタン:
　チェックサムを計算し、**挿入するアドレス**で指定されたバッファに書込まれます。

これらの設定はプロジェクトファイルには保存されません。これらは新しいデバイスを選択した時にイニシャライズされます。

メイン画面のチェックサムタブ

メイン画面に表示するサムの選択とそのチェック範囲を指定します。また一部のデータをチェック計算から除外する場合、ここで指定します。

・メイン・チェックサム用のアドレス範囲指定部:
　開始アドレス／終了アドレス:
　チェック Box ″Enable″ 時:チェックサムを計算するアドレス範囲が指定出来ます。アドレスは Byte アドレスで入力して下さい。
　チェック Box が ″Disable″ 時:設定したデバイスのディフォルトアドレスがサム計算に使用されます。
・メイン画面に表示するチェックサムの種類指定部:
　メイン画面に表示するチェックサムの種類をここで選択します。
・チェックサム　Result 表示部:
　最新のチェックサム値が表示されます。
・チェックサム計算から除外するバッファ指定部:
　一部のデータをチェックサム計算から除外設定する場合に使用します。
　ダイレクトサム計算タブ内の ″チェックサムから除外するバッファ指定″ と同じです。
・Apply　ボタン:
　Apply　ボタンを押すと、ここで設定したチェックサム設定が、デバイス書込み時の設定として採用されます。
　今まで使用していた設定が変更されてしまいますので、再確認してからボタンを押して下さい。

これらの設定はプログラマのコンフィグレーションファイルとプロジェクトファイルに保存されます。

各チェックサムの説明

・Byte sum (x8)
　使用するデバイスのデータ構成(x8/x16/x1)に関わり無く、常に 1byte 毎の単純加算値です。
　このモードを示す表示として「x8」をメイン画面のアドレス表示 Window のチェックサム部分に表示します。
・Word sum Little Endian (x16)
　使用するデバイスのデータ構成(x8/x16/x1)に関わり無く、常に 1word 毎の単純加算値です。
　サム計算するデータはバッファから Little Endian mode でリードします。
　このモードを示す表示として「x16 LE」をメイン画面のアドレス表示 Window のチェックサム部分に表示します。
　使用するデバイスのデータ構成(x8/x16/x1)に関わり無く、常に 1word 毎の単純加算値です。

・Word sum Big Endian (x16)

サム計算するデータはバッファから Big Endian mode でリードします。
このモードを示す表示として「x16 BE」をメイン画面のアドレス表示 Window のチェックサム部分に表示します。

・CRC-CCITT

バッファデータ(1byte)を 16bit WORD で計算する CRC-CCITT アルゴリズムを使用して結果を表示します。

使用する多項式= (x^16 + x^12 + x^5 +1)、初期値=0、XOR out=0、Reflection in/out=off

・CRC-XMODEM

バッファデータ(1byte)を 16bit WORD で計算する CRC アルゴリズムを使用して結果を表示します。

使用する多項式= (x^16 + x^15 + x^2 +1)、初期値=0

・CRC-16

バッファデータ(1byte)を 16bit WORD で計算する 標準 CRC-16 アルゴリズムを使用したサム値です。

使用する多項式= (x^16 + x^15 + x^2 +1)、初期値=0、XOR out=0

・CRC-32

バッファデータ(byte)を 32bit DWORD で計算する 標準 CRC-32 アルゴリズムを使用したサム値です。

使用する多項式は 0x04C11DB7 で、初期値=0xFFFFFFFF、XOR out=0xFFFFFFFF

・MD5

MD5 Hash は 32 桁の Hex 数で表示される 128bit のサム値です。

・SHA-1

″Secure Hash Standard″ は 40 桁の Hex 数で表示される 160bit のサム値です。

チェックサムの反転値　及び補数値

・SUM 値（Straigt）

ユーザーが指定したチェックサムで計算した結果をそのまま表示します。

・反転値（Nagated）

チェックサムを反転して表示します。
SUM 値 ＋ 反転値 = FFFFh になります。

・補数値（Supplement）

チェックサムの補数を表示します。
SUM 値 ＋ 補数値 = 0h になります。

デバイス依存のチェックサム

使用するデバイスによってチェックサム計算が決まっている場合があります。
例：STMicroelectronics STM8 ファミリー

チェックサムモードはプログラムウインドウ中のチェックサム表示部分をクリックするとチェックサム専用メニューが表示されます。
この専用メニューまたはショートカットでチェックサムを切り換えることも出来ます。
ショートカットは
Shift+Ctrl+1 は Byte sum (x8)
Shift+Ctrl+2 は Word sum Little Endian (x16)
Shift+Ctrl+3 は Word sum Big Endian (x16)
etc...
が選択出来ます。メインメニュー内のチェックサムオプションでも選択出来ます。

デバイス（メインメニュー・コマンド）

書き込むデバイス選択とデバイスの操作等を集めたメニューです。

デバイス->デバイス選択／履歴

最近使用したデバイス履歴リストの中からデバイスを選択出来ます。
このリストは最近使用したデバイスまたは選択したデバイスを20個まで記録しています。
このリストには選択したデバイス名とそのデバイスに設定したオプション情報も記録しています。
このリストは終了コマンド ファイル -> 保存して終了 実行時に自動的に記録されます。
現在使用しているデバイスの詳細情報を知りたい場合は、＜Ctrl＋F1＞keyまたは**"デバイス関連情報"**ボタンを押してください。このコマンドはデバイスのサイズ、構成、プログラミング・アルゴリズム、パッケージ情報などのデバイス情報を表示します。
リスト上の不要なデバイスは＜Del＞keyで削除出来ます。

デバイス->デバイス選択

サポートされている全てのデバイスの中から使用するデバイスを選択します。
ALLデバイス、デバイス種類から選択又は、メーカー名から選択を使用してデバイスを選択することが可能です。
Note 1:プログラマで表示しているデバイス名はフルNAMEではありません。
デバイス名称には温度用コード、スピード用コード、パッケージ用コード等の様々な文字が使用されていますが、プログラマの書込みでは必要ない場合があります。 これらの必要ない文字をデバイス選択画面では"xx"と表示しています。
使用するデバイスの捺印名やデータシートのパーツ名称部分を参考にしてデバイスを選択して下さい。

サンプル例：
・デバイス Am27C512-150,Am27C512-200,Am27C512-250 これらのデバイスは Am27C512 として登録しています。
・デバイス S29GL064N11TFC01 このデバイスは S29GL064NxxTxx01 として登録しています。

Note 2:同一名称のデバイスが 2 個表示されて、そのうちの一方の末尾に x16 が表示されている場合、これは書込み時間の速い WORD アルゴリズムを採用している事を表します。

選択したデバイスは 最近使用したデバイス履歴リストに登録されます。
この登録デバイスはデバイス -> デバイス選択/履歴 で使用できます。

検索フィールドに検索文字を入力することによって、選択するデバイスを絞り込むことが出来ます。
検索文字の間に"スペース"を入れることにより、検索条件のAND検索も可能ですので、見つけやすくなります。

現在使用しているデバイスの詳細情報を知りたい場合は、**デバイス情報** または ＜Ctrl+F1＞ key 押してください。このコマンドはデバイスのサイズ、構成、プログラミング・アルゴリズム、パッケージ情報などのデバイス情報を表示します。

デバイス–>デバイス選択–>ALLデバイス

サポートされている全てのデバイスの中から使用するデバイスを選択します。
サポートされているデバイス名はリストボックスに表示されますので、使用するデバイス表示名をダブルクリックします。
表示されるデバイス名が多すぎる場合は、検索ボックスにメーカー名とデバイス型番の一部を“スペース”で区切って入力すると表示数が絞り込まれ、選択しやすくなります。

ここで選択したデバイスはデバイス履歴リストに登録されます。
デバイス選択を中止する場合は＜ESC＞keyまたは**“Cancel”**ボタンを押してください。

選択する前に表示デバイスの詳細情報を知りたい場合は、＜Ctrl＋F1＞keyまたは**“デバイス関連情報”**ボタンを押してください。デバイスのサイズ、構成、プログラミング・アルゴリズムを表示します。パッケージ情報とデバイス情報も表示します。

デバイス–>デバイス選択–>デバイスの種類から選択

デバイスの種類を指定してから使用するデバイスを選択します。Subtypeでデバイスの容量別の分類を行なうことも出来ますので、さらに選択デバイスを絞り込むことが出来ます。
デバイス名がリストボックスに表示されますので、使用するデバイス表示名をダブルクリックします。
表示されるデバイスが多すぎる場合は、検索ボックスにメーカー名とデバイス型番の一部を“スペース”で区切って入力すると表示数が絞り込まれ、選択しやすくなります。

ここで選択したデバイスはデバイス履歴リストに登録されます。
デバイス選択を中止する場合は＜ESC＞keyまたは**“Cancel”**ボタンを押してください。

選択する前に表示デバイスの詳細情報を知りたい場合は、＜Ctrl＋F1＞keyまたは**“デバイス関連情報”**ボタンを押してください。デバイスのサイズ、構成、プログラミング・アルゴリズムを表示します。パッケージ情報とデバイス情報も表示します。

デバイス–>デバイス選択–>メーカー名から選択

デバイスメーカーを指定してから使用するデバイスを選択します。
最初にメーカー名リストボックスに表示されるメーカー名を選択します。選択したデバイスメーカーのデバイス名がリストボックスに表示されますので、使用するデバイス表示名をダブルクリックします。
表示されるデバイスが多すぎる場合は、検索ボックスにデバイス型番の一部を入力すると表示数が絞り込まれ、選択しやすくなります。

ここで選択したデバイスはデバイス履歴リストに登録されます。
デバイス選択を中止する場合は＜ESC＞keyまたは**“Cancel”**ボタンを押してください。

選択する前に表示デバイスの詳細情報を知りたい場合は、＜Ctrl＋F1＞keyまたは**“デバイス関連情報”**ボタンを押してください。デバイスのサイズ、構成、プログラミング・アルゴリズムを表示します。パッケージ情報とデバイス情報も表示します。

デバイス–>デバイス選択／デバイス ID

ソケット上のデバイス_ID情報をリードして、対応するデバイスを自動的に選択するコマンドです。

EP－ROM、FLASH－ROMはシグネチャーと呼ばれるIDコードを持っています。このIDコードには各デバイスメーカーに割り当てられたメーカーコードと、各メーカーが独自に付けたデバイスコードが書かれています。
プログラマはこのIDコードを利用してソケット上のデバイスを自動的に設定します。
ソケット上のデバイスがIDコードをサポートしていないときはUnkown又は、Unsupported Deviceエラーを表示します。
また対応するデバイス候補が複数検出された場合は、複数のデバイス名をリスト上に表示します。
使用するデバイスをこのリストで選択してください。
デバイス選択を中止する場合は**＜ESC＞**keyまたは**“Cancel”**ボタンを押してください。

警告　この自動設定は、EPROMのうち28pin、32pinデバイスのみ対応しています。
プログラマはIDコードを読むためにソケット上の対応ピンに高電圧に出力します。
EPROM以外のデバイスをソケットに装着しないで下さい。プログラマが高電圧を出力しますので、未対応デバイスにダメージを与えることがあります。

デバイス->デバイスオプション

デバイスの書き込みに関連する操作を集めたメニューです。

デバイス->デバイスオプション->動作オプション

デバイスの動作範囲設定やコンタクトチェックの有無、書き込み前のErase、Blankの有無、デバイスのコンフィグレーション設定等を指定します。

＊　ここでの設定項目は使用するデバイスにより異なります　＊

デバイス選択時にはほとんどの設定はDisable、またはディフォルト値になっています。
これらの設定は選択されているデバイス情報と一緒に ファイル->保存して終了コマンドにより保存されます。

デバイス動作オプション設定項目

・Device／Bufferアドレス グループ

Device スタート	:デバイスリード／ライトする開始アドレス（ディフォルト 0）
Device エンド	:デバイスリード／ライトする終了アドレス （ディフォルト デバイスサイズー1）
Buffer スタート	:バッファ開始アドレス （ディフォルト 0）
Buffer分割	:16bit／32bitデータを 8bitに分割する方法 （ディフォルト 分割設定なし）

・Buffer分割の詳細

このオプション設定は16bit／32bitアプリケーション用に作成したデータを8bitデバイスに書き込む場合、作成したデータのどの部分を使用するか指定します。

	Buffer 分割 type	Device アドレス <==> 使用されるBufferアドレス
1	None	[ADDR+0] <==> Buffer[ADDR]
2	Even	[ADDR+0] <==> Buffer[0 +(2＊ADDR)]
3	Odd	[ADDR+0] <==> Buffer[1 +(2＊ADDR)]
4	1／4	[ADDR+0] <==> Buffer[0 +(4＊ADDR)]
5	2／4	[ADDR+0] <==> Buffer[1 +(4＊ADDR)]
6	3／4	[ADDR+0] <==> Buffer[2 +(4＊ADDR)]
7	4／4	[ADDR+0] <==> Buffer[3 +(4＊ADDR)]

デバイスにリード／ライトする時のバッファアドレス 対応表

デバイスにリード／ライトした時の実際のバッファアドレス例
上記の表でADDR=0にした時の例です。

	Buffer 分割 type	上段:Device アドレス 下段:Buffer アドレス
1	None	Device address 00 01 02 03 04 05 06 07 08 09 … Buffer address 00 01 02 03 04 05 06 07 08 09 …
2	Even	Device address 00 01 02 03 04 05 06 07 08 09 … Buffer address 00 02 04 06 08 0A 0C 0E 10 12 …

3	Ｏｄｄ	Device address　00 01 02 03 04 05 06 07 08 09 … Buffer address　01 03 05 07 09 0B 0D 0F 11 13 …
4	１／４	Device address　00 01 02 03 04 05 06 07 08 09 … Buffer address　00 04 08 0C 10 14 18 1C 20 24 …
5	２／４	Device address　00 01 02 03 04 05 06 07 08 09 … Buffer address　01 05 09 0D 11 15 19 1D 21 25 …
6	３／４	Device address　00 01 02 03 04 05 06 07 08 09 … Buffer address　02 06 0A 0E 12 16 1A 1E 22 26 …
7	４／４	Device address　00 01 02 03 04 05 06 07 08 09 … Buffer address　03 07 0B 0F 13 17 1B 1F 23 27 …

説明：Oddモードで8bitデバイスをリード／ライトした場合
デバイスの0，1，2，3，4･･･番地には、バッファメモリの1，3，5，7，9･･･番地のデータがリード／ライトされます。

・デバイスチェック　グループ

デバイス端子チェック（ディフォルト　デバイスチェック有り）
チェック有り設定の場合、プログラマはZIFソケットに挿入されたデバイスの接触不良をチェックします。このチェックにより、デバイスの逆挿し、位置ズレ等の誤挿入をチェックします。

デバイス　ID　チェック（ディフォルト　IDチェック有り）
IDチェック有り設定の場合、Read、Blank、Verify、Program、Erase等の動作開始前にZIFソケット上のデバイスのIDチェックを行ないます。

注意：ID機能をサポートしていないデバイスもあります。
この場合ID　Checkは自動的にDisableになり、IDチェックは行ないません。

・プログラマ動作指定　グループ

Program前にErase　（ディフォルト　Disable）
Enable時：Program前にErase（消去）を実行する

Program前にBlank　（ディフォルト　Disable）
Enable時：Program前にBlank（未書き込みチェック）を実行する

Read後にVerify　（ディフォルト　Enable）
Enable時：Read後にVerify(データ確認)を実行する

Verify回数
Verify回数（Once：1回、Twice：2回）を指定する。

Verify options (normal VCC +/- 5% normal VCC +/- 10% VCCmin -- Vccmax)
Verify回数がTwice（ 2回 ）の場合のVcc電圧を指定する。

・Programmingパラメータ
プログラムするエリア、ブロック等の指定、またはチップのプロテクト設定等の指定に使用します。

・Eraseパラメータ
消去機能を持ったデバイス選択時に表示されます。消去する範囲やエリア等の指定を行う時に設定します。

・ターゲット・システム用電源パラメータ　関連設定
この設定は ISP モード対応のデバイスを選択した時に変更可能になるパラメータです。

Enable ターゲットに電源を供給する

Enable 設定時:ターゲット ISP デバイスにプログラマ側から電源を供給します。

電源は書き込み開始時に ON し、書き込み完了時に OFF します。

但し”動作完了後 ISP 信号を設定したレベルに保持する”が設定されている場合は、信号レベル Pull-UP / Pull-Down が OFF した 時に、供給電源を OFF します。

電圧

ターゲット ISP デバイスへの供給電源電圧を設定します。設定可能範囲は 2V to 6V です。

Note:

ターゲット ISP に供給する電源電圧は、ターゲット ISP が消費する電流によって影響を受けます。ターゲット ISP に供給する電圧と最大電流はなるべく正確な値を設定してください。

最大電流(未使用)

ターゲット ISP デバイスの最大消費電流を設定します。設定範囲は 0 to 300mA。

電源の立ち上がり時間

ターゲット用供給電源の立ち上がり時間を指定します。

ターゲット電源のセット時間

ターゲット ISP デバイスに供給した電源が設定電圧で安定し、しかも ISP デバイスの書き込みが可能になるまでの時間を設定します。

電源の立ち下がり時間

ターゲット用供給電源の立下り時間を指定します。

ターゲット電源の OFF 時間

ターゲット ISP デバイス電源を OFF 後、コンデンサ等にチャージされている電圧が放電されるまでの時間を設定します。

デバイス電源が完全に OFF し、ISP ケーブルを安全に抜き取ることができる時間を設定してください。

・ターゲット・システム用パラメータ 関連設定

この設定は ISP モード対応のデバイスを選択した時に変更可能になるパラメータです。

Oscillator frequency (xxxx Hz)

ターゲットデバイスで使用するクロック周波数を指定します。

プログラマ側から設定周波数を出力しますので、正確な値を設定して下さい。

ターゲット側の電源電圧

ターゲット・システム内の電源電圧を指定します。プログラマはデバイスのリード/ライト動作前にこの電圧をチェックします。

電源電圧(xxxx mV)

プログラマがチェックするターゲット System 側の電源電圧を指定します。

ターゲット側電圧をチェックしない

リード/ライト動作開始前の電圧チェックを Disable にします。

RESET解除後の遅延時間

リセット信号がアクティブになってから、ターゲットデバイスの書き込みを開始するまでの時間を設定します。

この時間は使用するデバイス、ユーザーのリセット回路により設定時間が異なります。

10ms, 50ms, 100ms, 500ms or 1s.の設定から選択できます。

待機時の ISP 信号レベル

デバイスを Disable 状態にする ISP 信号レベルを指定します。

Pull-up(22k でプログラマ内部の電源に接続)または Pull-down(22k でプログラマ内部の GND に接続)を選択します。

動作完了後 ISP 信号を設定したレベルに保持する 設定

Enable 設定時:プログラマは ISP 書き込み終了後も、 ISP 信号レベルを Pull-UP / Pull-Down 状態に保持します。

また ISP 信号が出力されていることを警告ウィンドウに表示します。ユーザーがウィンドウを閉じたときにプログラマは ISP 信号を OFF します。

デバイス->デバイスオプション->シリアライズ

シリアライズは、個々のデバイスに自動的に番号を書き込む機能です。
シリアライズモードを起動した場合、各デバイスのProgram前にプログラマのバッファメモリ上にユーザーが希望する番号を自動的に挿入します。この機能によってユーザーは個々のデバイスに異なった番号を書き込むことが出来ます。

シリアライズには、3種類のモードがあります。

- ・インクリメントモード
- ・ファイル参照モード
- ・カスタムジェネレートモード

またこのダイログにはプロジェクトファイルからシリアライズを使用する時に必要となる関連付けされたファイル(シリアライズポジションファイル)を設定するための項目もあります。プロジェクトファイルでシリアライズを使用する方法の詳細はシリアライズとプロジェクトを参照して下さい。

シリアライズ機能の基本的なルール
新しいデバイスが選択された際には、シリアライズ機能は無効になります。
シリアライズ機能を使用して書き込んだ場合、ファイル->保存して終了 操作によりシリアライズ情報は保存されます。

インクリメントモードを選択した際には、シリアライズ情報(アドレス、サイズ、シリアル番号、ステップ値)と書き込む番号の書式(ASCII/BIN、DEC/HEX、LS byte/MS byte)が保存されます。

ファイル参照モードを選択した際には、シリアライズ情報(シリアル番号を記録したファイル名、使用したシリアル番号に付いているラベルや行番号)が保存されます。

マルチプログラマモードで使用する時
書込みデバイスでエラーが発生した場合に使用されなかったシリアライズ番号を処理する方法が指定出来ます。

1. プログラム出来なかったシリアル番号を無視する。
2. プログラム出来なかったシリアル番号をファイルに追記する。

'プログラム出来なかったシリアル番号を無視する' を選択した場合、
このシリアル番号は無視されます。また通常の動作しか行いません(特別なアクションは発生しません)
'プログラム出来なかったシリアル番号をファイルに追記する' を選択した場合、
プログラム出来なかったシリアル番号をファイルに追記します。
この追記したファイルには、ファイル参照モードで使用しているフォーマットで記録しますので、 最後にファイル参照モードを使用してプログラム出来なかったシリアル番号を再利用することが出来ます。

もしユーザーによって、書き込みが中止された場合、シリアル番号は使用されず、次のデバイスに使用されます。
書き込みが正常終了しない場合もシリアル番号は使用されません。例デバイスのコンタクトチェックエラー時がこれに相当します。

シリアライズ機能はプログラマ内の main_buffer を使用したり、また一部のデバイスではシリアル番号機能に利用可能な拡張メモリを使用します。例:Microchip社のPIC16Fxxx では Data EEPROM Memoryが利用出来ます。

どの方法を利用するかは、シリアライズ設定ダイアログ画面で指定します。
デバイス内の拡張メモリを使用する方法を選択する場合は'ファイル参照モード'は使用出来ません。
利用上の制限事項等の詳細はファイル参照モード側をご覧下さい。

チェックサムについて: シリアライズ機能を使用した場合、書き込んだデバイス毎にチェックサム値が異なってしまいます。シリアライズ番号に使用するバッファをあらかじめチェックサムの計算範囲から除外する機能があります。設定は**バッファチェックサム** の**メイン画面のチェックサム**を参照して下さい。

デバイス−>デバイスオプション−>シリアライズ−>インクリメントモード　&　SQTP

各々のデバイスに個別のシリアル番号を書き込みます。
書き込むシリアル番号は最初のスタート番号、シリアル番号のSTEP値、書き込むアドレスそして書き込むフォーマットを指定できます。
またこのオプション設定はMicrochip社製PICmiconの同様な機能(Microchip　SQTP)もサポートしています。
次のオプションが使用可能です。

S/N　サイズ指定

S／N　サイズは、書き込むシリアル番号の桁数をバイト方式にて指定します。
バイナリーフォーマットでは1～8バイトが有効です。
アスキーフォーマットでは1～16バイトが有効です。。

Address指定

シリアル番号を書くバッファメモリアドレスを特定します。
デバイスのスタートアドレスとエンドアドレス範囲に注意してください。
シリアル番号が、スタートアドレスとエンドアドレスに入るように指定してください。

スタート番号

最初のデバイスに書き込むスタート番号を指定します。
使用できる値は　0x0——0x1FFFFFFF
超えた場合は0x0にもどります。

ステップ

ステップは、段階的に増加するシリアルの数値を指定します。

S/N　mode

S／N modeは、デバイスに書くシリアル番号をASCII形式／BIN形式のどちらの形式を使用するか指定します。

例：デバイスに書くシリアル番号＝0x0528CD　の時
・ASCII指定　：ASCII文字で記録します。
プログラマのバッファに0x30、0x35、0x32、0x38、0x43、0x44が上書きされ、このデータがデバイスに書かれます。
（デバイスには6バイト‘0’、‘5’、‘2’、‘8’、‘C’、‘D’が書かれます。）

・BIN指定　：バイナリーで記録します。
プログラマのバッファに0x05、0x28、0xCDが上書きされ、このデータがデバイスに書かれます。
（デバイスには3バイト0x05、0x28、0xCDが書かれます。）

Style

シリアル番号の表現スタイルを指定します。下記の二つの方式が選択できます。
シリアル番号のスタート値、ステップ値を指定する前にこのシリアル番号のStyleを指定してください。

・Decimal／10進数：0から9までの数値を使用して表示します。
・Hexdecimal／16進数：0からFまでの数値を使用して表示します。

番号の並び方

シリアル番号をプログラマのバッファに上書きする時の数値の並び順を指定出来ます。
この設定は、バイナリー・シリアルモード時に使用出来ます。

・LS_Byte first:（主にインテルプロセッサーで使用される）は、バッファ内の下位アドレスに最小バイトのシリアル番号を上書きします。

・MS_Byte first:(主にモトローラープロセッサーに使用される)は、バッファの下位アドレスに最大バイトのシリアル番号を上書きします。

シリアル番号の分割

このオプションは、シリアル番号を分割し、バッファ上に1～3byteの間隔を置きながら分割したシリアル番号を配置します。
この方式は、特にMicrochip PICデバイスSQTP serialization modeに有効です。
（メモリに書きこまれたシリアル番号をプログラムの一部として使用する場合に有効です。
詳細は例2を参照して下さい）

例1: シリアル番号を AT29C040 のデバイスの 0x7FFFA に書き込む。
シリアル番号は、4Byte、スタートの値は 0x16000000, incremental ,ステップは 1、シリアル番号の形式はバイナリー、シリアル番号の LSB バイトはデバイスの下位アドレスに置く。
上記の状態にするには、下記の設定にシリアライズダイアログを設定しなければならない。

Mode　:　インクリメント
S/N サイズ　:　4Byte
Address　:　0x7FFFC
スタート番号　:　0x16000000
ステップ　:　1
S/N mode　:　BIN
Style　:　Hex
番号の並び　:　LS Byte first
シリアル番号の分割:チェック無し

下記のように書き込まれます。

・最初のデバイス
Address　Data
007FFF0　xx xx xx xx xx xx xx xx xx xx xx xx 00 00 00 16
・2番目のデバイス
Address　Data
007FFF0　xx xx xx xx xx xx xx xx xx xx xx xx 01 00 00 16
・3番目のデバイス
Address　Data
007FFF0　xx xx xx xx xx xx xx xx xx xx xx xx 02 00 00 16
: :
シリアル番号のサイズが4バイトの為、0x7FFFC から 0x7FFFF に書きこまれる。

例　2：次の例題は、SQTP serialization modeの使用方法です。
Example 2-a：RETLW インストラクションを使用したシリアル番号管理方法、
Microchip PIC16F628
Example 2-b：NOP インストラクションを使用したシリアル番号管理方法、
Microchip PIC16FJ256

例　2-a：RETLW インストラクションを使用したシリアル番号管理方法、Microchip PIC16F628

PIC 16F628 は、命令長 14bit , RETLW　は 14bit オペコードです。

Description		MSB 14-Bit word LSB
RETLW	Return with literal in W	11　01xx　kkkk　kkkk

PIC デバイスの 4 個の RETLW 命令にシリアル番号 0x1234ABCD を書き込むことを想定します。
最上位のシリアル番号が MSB になります。デバイスの 0x40 番地にシリアル番号を書き込む。
この場合シリアル番号の分割モードが最も有益です。シリアライズはシリアル番号を分割する
ことなく、次の番号をバッファとデバイスに書き込むことができます。

Address　Data
0000080　CD AB 34 12 xx xx xx xx xx xx xx xx xx xx xx xx

注意：0x80 は、バッファがバイト方式で PIC デバイスがワード方式なのでメモリアドレスは 0x40
になります。

データ 0x1234ABCD は次のようにバッファ入る
Address　Data
0000040　ABCD 1234 xxxx xxxx xxxx xxxx xxxx xxxx

RETLW 命令を使用したい場合はバッファには下記のデータを入れます。
Address　Data
0000040　34CD 34AB 3434 3412 xxxx xxxx xxxx xxxx

このデータは下記の方法で作成できます。
2-a-1. メインバッファの0x40に4つのRETLW命令を書く。(これはバッファの編集またはファイル
の読み込みによって実行できる)
それぞれのRETLWボトムの8ビットは、現時点で重要ではない。なぜなら、シリアライズは、ボト
ムの8ビットにRETLW命令の8ビットの正しいシリアル番号を書く。
PICデバイス書き込み前のバッファの中身は、下記のようになる。

Address　Data
0000040　3400 3400 3400 3400 xxxx xxxx xxxx xxxx

それぞれのRETLW命令の8ビットは、ゼロになっていますが、任意の値をもつことができます。

2-a-2. シリアライズダイアログに下記の設定をします。

S/N　サイズ	:	4Byte
Address	:	40H
スタート番号	:	1234ABCDh
ステップ	:	1
S/N mode	:	BIN
Style	:	HEX

番号の並び　　　　：　LS byte first
シリアル番号の分割：　チェック有り
番号と番号の間　　：　1 byte(s)
S/N 分割サイズ　　：　1 byte(s)

シリアル番号の分割がすべて2バイトでバッファされることを意味しています。
プログラマの動作前、ここで説明したデータがバッファ内に設定されます。

最初のデバイスに書く前のバッファの中身のシリアル番号は

Address　Data
0000040　34CD 34AB 3434 3412 xxxx xxxx xxxx xxxx

2番目のデバイスには

Address　Data
0000040　34CE 34AB 3434 3412 xxxx xxxx xxxx xxxx

この次のデバイスも +1 された番号が書かれます。

例 2-b:NOP インストラクションを使用したシリアル番号管理方法、Microchip PIC24FJ256

PIC 24FJ256 は、命令長 24bit、コード 00xxxxh の NOP インストラクションを持つデバイスです。
Microchip MPLAB の SQTP シリアル番号と同一の番号管理を指定出来ます。

以下のステップでこの番号管理が実行出来ます。

2-b-1. プログラマの Buffer アドレス=800h に NOP インストラクション(00xxxxh)を書きます。この作業はマニュアル EDIT 操作またはこのデータが書かれたファイルを読込みます。
プログラマの 800h 番地は PIC24Fxxx デバイスの 200h 番地に相当します。(詳細はデバイスのデータシートを参照して下さい。

プログラム開始前の 800h 番地の Buffer 内容は

Address　Data
0000800　00 00 00 00 00 00 00 00 xx xx xx xx xx xx xx xx

2-b-2. シリアライズダイアログに下記の設定をします。

S/N サイズ　　　　：　3Byte
Address　　　　　：　800H
スタート番号　　　：　123456H
ステップ　　　　　：　1
S/N mode　　　　：　BIN
Style　　　　　　：　HEX
番号の並び　　　　：　LS byte first
シリアル番号の分割:チェック有り
番号と番号の間　　：　2 byte(s)
S/N 分割サイズ　　：　2 byte(s)

シリアル番号は 2byte 単位に分割されたバッファに設定されます。
プログラマの動作前、ここで説明したデータがバッファ内に設定されます。

最初のデバイスに書く前のバッファの中身のシリアル番号は

Address　Data
0000800　56 34 00 00 12 00 00 00 xx xx xx xx xx xx xx xx

2 番目のデバイスには

Address　　Data
0000800　　57 34 00 00 12 00 00 00 xx xx xx xx xx xx xx xx

この次のデバイスも +1 された番号が書かれます。

例3. 例 2-a と同様なシリアル番号オプション設定ですが、シリアル番号書き込みアドレス間隔を2、3 個にした例です。

"挿入する間隔 2byte"を設定, このときのバッファの中は
元のデータ:Byte 表示時
Address　　Data
0000080　　CD xx xx AB xx xx 34 xx xx 12 xx xx xx xx xx xx
元のデータ:Word 表示時
Address　　Data
0000040　　xxCD ABxx xxxx xx34 12xx xxxx xxxx xxxx

"挿入する間隔 3byte"を設定, このときのバッファの中は
元のデータ:Byte 表示時
Address　　Data
0000080　　CD xx xx xx AB xx xx xx 34 xx xx xx 12
元のデータ:Word 表示時
Address　　Data
0000040　　xxCD xxxx xxAB xxxx xx34 xxxx xx12 xxxx

アドバイス:

シリアライズの効果について確信がない時、実際に書かれる シリアル番号(バッファに書き込まれる番号)を前もってテストすることが可能です。
下記にテスト方法を記述します。
1. ダイアログシリアライズ内のシリアル番号設定を行い、OK ボタンを押す。
2. **デバイスオプション -> 動作オプション**設定内のインサーションテストとデバイス ID チェックを Disable(チェックしない)にセットする。
3. ZIF ソケット内にデバイスがないことを確認
4. **Device Program** を起動(いくつかのタイプのデバイスは、プログラマを起動する前にプログラミングオプションを選択する必要あり)
5. 書き込み動作が完全に終わった後、(デバイスを挿していないので、大半のデバイスはエラーを表示する)
シリアル番号が書き込まれているメインバッファを確認してください。(バッファの表示/編集)

注意:

シリアル番号が書かれるアドレスは、書き込みを行なうデバイスの構成により影響します。
バッファ構成が、(X8) のバイト構成だったら、シリアライズのアドレスは、バイトアドレスになります。もし、バッファ構成がバイトより(例えば 16 ビットワード)より広ければ、シリアライズアドレスは、ワードアドレスになります。

デバイス->デバイスオプション->シリアライズ->ファイル参照　モード

ユーザーが用意したシリアル番号をファイルから読込み、指定したアドレスに書き込みます。
使用できるファイルフォーマットは下記の2種類の方式があります。

（1） Classic From-file mode

シリアル番号を直接入れたファイルを使用します。
ファイルの中で指定されたアドレスが参照されて、バッファメモリ上のこのアドレスに対応する場所にファイルデータが読み込まれます。
このモードは、メイン画面上のアドレス(hex)表示パネル内のシリアライズ部分に ″ファイル参照″ と表示されます。
Classicについての記述は ″**Classic From-file serialization file format**″の章に記載があります。

（2） From-file mode

シリアル番号以外の任意のデータをデバイスに書きこむ場合に“playlist”を使用します。
ファイルの中身は、シリアライズデータが格納されている外部ファイル名がリストとして記入されています。
この外部データファイルから書き込むデータを読み込みます。それぞれのファイルは、1個のデバイスに対応します。
このモードは、メイン画面上のアドレス(hex)表示パネル内のシリアライズ部分に″ファイル参照-pl″と表示されます。
詳細記述は、**″Playlist　From-file serialization file format″**の章に記載があります。

PG4UW ソフトはユーザーが使用するファイル内のデータ構成から、自動的に **Classic From-file mode** か **From-file mode** を判断しますので、ユーザーがモードを指定する必要はありません。

上記のファイルモードには二つのオプション設定があります。

File name

ファイルネームオプションは、シリアル番号と書く込むアドレス情報を読み込むファイル名を指定します。シリアライズファイルで使用できるファイルは、下記の **From-file serialization file format** で記述された特別のフォーマットです。

Start label

上で指定したファイル内に記入されているスタートラベルを指定します。シリアル番号の読み込みはここで指定したスタートラベルから行ないます。
シリアライズファイルのサイズは、ディスクの容量によって制限され、推奨値は最大シリアルレコード数 10000 レコードです。
それ以上のレコード数は、シリアル番号読み込みの際に動作遅れの原因となります。

ファイル参照モード(Playlist)を指定した時に有効となる　指定項目

使用したファイルの追加動作

ユーザーが Playlist From-file mode を使用した場合、下記の3種類のファイル操作から1個を選択できます。

・何もしない
　－ 使用したシリアル番号ファイルはそのままにする。
・使用したファイルを他のディレクトリへ移動
　－ 使用したシリアル番号ファイルを指定したディレクトリへ移動する。
・使用したファイルを削除
　－ 使用したシリアル番号ファイルを削除する。

ディレクトリ

ファイル参照モード内の **使用したファイルを他のディレクトリへ移動**を選択した時に有効になります。この部分に移動するディレクトリを入力して下さい。

ファイル参照モード(Playlist)を使用した時に発生するエラー番号について

・S/N error #3 – "シリアル番号ファイルが見つかりません。"
・S/N error #34 –"使用したシリアル番号ファイルが削除できません。"
ファイルにプロテクトが掛けられている場合があります。
・S/N error #35 –"使用したシリアル番号ファイルが移動できません。"
ファイルにプロテクトが掛けられているか、または移動先のディレクトリがありません。

・Classic From-file モードで使用するファイルフォーマット

Classic From-file serialization で使用するファイルは、テキストフォーマットです。
このファイルは、バッファに書き込むアドレスとシリアルデータを含んでいます。

入力ファイルは、テキストフォーマットで構造は

［Label1］ addr byte0 byte1．．．byteN
［LabelN］ addr byte0 byte1．．．byteM ， addr byte0 byte1．．．byteK
basic part **optional part**

上記解説：

・Label1，Label2．．．LabelS

ラベルは、入力ファイルのそれぞれのラインを特定します。それぞれのラインの区切りに使用されますので、各々異なったラベル名称をつけます。
またこのラベルはどのシリアル番号から使用するのか判断するときに利用されます。

・basic part

ベーシックパートは、バッファアドレスとバッファに書き込むデータ（バイト）配列を定義します。
ベーシックパートは先頭のラベルの後に定義します。

・optional part

オプショナルパートは、2番目のバッファアドレスとバッファに書き込むデータ（バイト）配列を定義します。オプショナルパートは、ベーシックパートの後に定義します。またoption partの区切りには"，"（カンマ）を使用します。

・addrー

データを書くバッファアドレスを指定します。

・byte0．．byteN ， byte0．．byteM ， byte0．．byteK

シリアル番号として書き込むデータをデータ配列にして指定します。
最大指定可能数は64バイトまでです。このデータはバッファのaddrーaddr＋Nに書かれます。

byte0 ー＞addr
byte1 ー＞addr＋1
byte2 ー＞addr＋2
：
byteN ー＞addr＋N

・使用する記号の説明

- ラベル名は小文字と大文字は同じ文字として解釈される。
- 小文字と大文字は同じ文字として解釈される。
- ″[　]″ - ラベルはカギカッコの中に書く。
- ″,″ - データの区切り文字。
- ″;″ - コメントマーク、このマークから後ろはコメントとして解釈される。
- アドレスとデータはすべてhex-decimalで記述する。
- 許容アドレスは、1から4バイトです。
- ひとつのラインの許容データは、1から64バイトです。
- ひとつのラインに、二つのデータ配列があるとき、許容データは合計で最大80バイトです。
- アドレスは、デバイスのスタートとエンドアドレスの範囲内で定義してください。
- アドレスが範囲外の場合、警告メッセージを表示し、シリアライズは、機能しなくなります。
- シリアライズのアドレスは、いつもデバイスの構造やバッファ構造にあわせて割り当てられます。

Example ：

```
[nav1]A7890  78  89  56  02  AB  CD; コメント1
[nav2]A7890  02  02  04  06  0A  25
[nav3]A7890  08  09  0A  0B  A0  C0; コメント2
[nav4]A7890  68  87  50  02  0B  8D
[nav5]A7890  A8  88  59  02  AB  7D
次の行は、2個のシリアライズを定義しています。
[nav6]A7890  18  29  36  42  5B  6D, FFFF6  44  11  22  33
                  99  88  77  66  55  16
        ;コメント  ここが最終のシリアル番号
```

この例では6個のラベルnav1．．．nav6を定義しています。
各々0xA7890番地から6バイトのデータをバッファに書きます。
nav6では追加のデータ配列を定義しています。追加の10個のデータはバッファアドレス
0xFFFF6－0xFFFFFに書かれます。

注意：シリアル番号書込みで使用されるアドレスは、使用するデバイスのデータ幅によって決まります。もしデバイスのデータ幅が 8bit の場合そのシリアライズのアドレスは、バイトアドレスになります。
また、デバイスのデータ幅が 16bit の場合、そのシリアライズのアドレスは、ワードアドレスとして使用されます。

・**Playlist From-file serialization モード**で使用するファイルフォーマット

From-file serialization playlist ファイルは、シリアル番号データが記入されたデータファイルを集めたリストファイルです。
そのファイルフォーマットは、Classic serialization file フォーマットに類似しています。
次項に Classic serialization file フォーマット との違いを列記します。

1. プレイリストファイルには、ファイル先頭部分にスペシャルヘッダーがあります。
   FILETYPE=PG4UW SERIALIZATION PLAYLIST FILE
   このヘッダーは、テキスト文字入力です。
2. それぞれのシリアルデータファイル名は、1 ライン毎に記入します。
   ［label *xx*］ datafile name

・ラベルは、入力ファイルの識別子です。
このラベルは、それぞれのファイルを識別するためのライン番号またはアドレス情報をつけるために使用されます。
このラベルは、ファイル内で各々異なった番号を使用します。
ラベル番号をつけているので、ユーザーはどのシリアル番号から使用するのか判りやすくなります。

・データファイルネーム
シリアライズデータを含むデータファイルの名前を定義します。
シリアライズが、新しいシリアル番号を要求する際、そのデータファイルは、PG4UW″ファイルの読込″の手順によりバッファに読み込まれます。
ファイルフォーマットは、Binary もしくは Hex ファイル（INTEL.Hex など）になります。

Example: Playlist

```
;―――つぎの行はヘッダーファイル―――
    FILETYPE＝PG4UW SERIALIZATION PLAYLIST FILE
;―――参照するシリアル番号ファイル（ラベル ファイル名）
    ［nav1］ files1．dat
    ［nav2］ files2．dat
    ［nav3］ files3．dat
        : :
    ［navN］ filesN．dat
;――― end of file ―――
```

From-file Playlist機能のより詳しい情報を知りたい場合は、下記のファイルを参照してください。
C:¥ProgramFiles¥minato¥M1883¥Programmer¥Examples
¥Serialization¥fromfile_playlist_example¥

また次の手順でシリアライズテストを実行出来ます。

1. .PG4UW を起動
2. プログラマ接続を確認します。
3. デバイスを選択します。テストには消去可能なデバイスが適しています（OTP メモリは、不可）
4. **デバイス -> デバイスオプション -> シリアライズ**を選択してください。
5. ファイル参照モード を設定し、パネル内のファイル参照モード オプション でシリアライズの例を PC 上のファイルから選択してください。
6. この Windows の **OK** ボタンをクリックしてください。
7. デバイス動作(Program 等)を実行してください。
   メイン Window と経過表示画面にシリアライズ項目部分に使用するラベルが表示されます。

デバイス–>デバイスオプション–>シリアライズ–>カスタムジェネレートモード

Custom generator モードは、融通性のあるシリアライズモードです。このモードではシリアル番号の管理方法をユーザー自身で構築できます。

Custom generator モードが選択された際、シリアルナンバーは、ユーザープログラムによって生成されますので、ユーザーが独自のシリアル番号を書き込むことができます。例えば連続したシリアル番号や、不規則なシリアル番号を書き込むことも出来ます。プログラムの詳細は、下記のセクション**Custom generator program**を参照してください。

Examples :

サンプルファイルはPG4UWをインストールしたディレクトリのExamples￥のサブディレクトリー

< PG4UW_inst_dir>¥Examples¥Serialization¥customgenerator_example¥

の中にあります。C：がrootディレクトリの場合一般的には

C:¥ProgramFiles¥MINATO¥M1883¥Programmer¥Examples¥Serialization¥
customgenerator_example¥

になります。

Custom generatorモードには、次のオプションがあります。
シリアライズダイアログ内の –>シリアライズ機能の設定 –> Custom generator modeを選択します。

Serialization Data File

使用するシリアルナンバーを含むファイルのパスとファイル名を指定します。
デバイスが書き込まれるとき、PG4UW のソフトは、データファイルを更新するためにユーザーが生成したシリアル番号生成ソフトをコールします。
データファイルの拡張子は、.dat を推奨します。この拡張子は BPMICRO 社のプログラマで使用しているシリアル番号生成ソフトとコンパチですので、シリアル番号用データファイルがそのまま使用できます。

注意： デバイスにシリアル番号を書いている間にデータファイルがアクセスされます。指定するデータファイル名及び拡張子".dat"を確認してください。

Serialization generator

シリアル番号生成ソフトのパスと実行ファイル名を指定します。

First serial number

このオプションに、シリアル番号の初期値を指定します。この番号はシリアル番号生成ソフトに送られます。Hex(16進数)フォーマットで指定して下さい。

Last serial number

このオプションは、シリアル番号の最大値を指定します。 Last serial numberがゼロ以外の場合、その数値をシリアル番号生成ソフト側に送ります。シリアル番号生成ソフトはこの数値をチェックし、最新のシリアル番号を作成して.datファイルに書き込みます。
Last serial numberが現在使用しているシリアル番号よりも小さい場合は、エラー情報を.datファイルに書き込みます。
Last serial numberがゼロの場合は、その数値はシリアル番号生成ソフト側に送られません。

チェックBox

デバイス書込み完了後、書込み結果を -RESULT パラメータでシリアル番号生成ソフト側に送る。(ディフォルト設定ではチェック無し)

・この Box をチェックした場合、-RESET パラメータがシリアル番号生成ソフトに送られます。プログラマでデバイスの書込みが完了すると、PASS/FAIL 結果がシリアル番号生成ソフトに送られます。

・動作結果パラメータの書式:

-RESULT[n]=TRUE | FALSE -N<s/n>

n はマルチプログラミング時のプログラマ接続サイト番号を示す数字です。1台動作では出力されません。

TRUE はデバイス書き込み PASS 終了を示し、FALSE はデバイス書き込み ERROR を示しています。

-N<s/n> パラメータは -RESULTが実行されたときに使用されるシリアル番号です。

Examples :

generator.exe -RESULT=TRUE -N1234.

プログラマでシリアル番号 '1234' の書込みが PASS した事を シリアル番号生成ソフト "generator.exe" にパラメータとして渡しています。

・Custom generator program

シリアル番号生成ソフトはユーザーが必要とする番号を生成し、シリアル番号を登録するデータファイル. datを作成します。

このシリアル番号生成ソフトはユーザーが自分の仕様にあったものを作成します。作成したソフトの名称とパス(ファイルの場所)はCustom generator modeオプションで指定します。

このソフトはPG4UWプログラマが新しいシリアル番号を必要とするときに、CALLされます。通常は各々のデバイスの書き込みを行なう前にCALLされます。

PG4UWは、コマンドラインのパラメータをシリアライズプログラムとシリアル番号生成ソフトに送ります。下記のコマンドラインパラメータが使用されます。

-N＜serial number＞は、現在のシリアルナンバーを指定する。

-E＜serial number＞は、最後の(最新の)シリアルナンバーを指定する。

このパラメータは、Last serial numberの値が0以外のときに送られます。

最新シリアル番号が最終番号よりも大きくなった場合、. dat fileにエラーコード "T06"が記録されます。詳細については、.dat file format のシリアライズのセクションを参照して下さい。

Serialization .dat file format

Serialization.dat ファイルは、シリアル番号生成ソフトによって下記のテキストフォーマットで作られます。Serialization.dat ファイルは、レコードとシリアルデータセクションから構成されます。

レコードは、下記のTxx(接頭)のひとつから始まる。

"xx"の数値は、レコードタイプコードを示す。

レコードは、PG4UWのSerialization 情報(現在と最終のシリアルナンバー、シリアライズデータ、データフォーマット、エラーなど)を示します。

必要なレコードは、T01,T02,T03,T04で他のレコードについては、オプションです。

T01:<serial number>
現在のシリアルナンバーをコマンドラインパラメータ '-N<serial number>' で生成ソフト側に送る。

T02:<serial number>
次回に使用するシリアルナンバー。プログラマは現在使用しているシリアル番号の次に使用する。

T03:<data format code>
Serializationのデータフォーマットを指定する。

下記のフォーマットが現在サポートされている。
T03:50 or T03:55 – ASCII Spaceフォーマット
T03:99 – Intel HEX / Intel Extended Hexフォーマット:max address=0xFFFF

T04:T04を記述した次のラインから、ファイルの最後までがシリアルデータとして使用されます。
データは、標準的なアスキーデータとして格納されます。
例えば、INTELHEX、ASCII SPACEなどです。

例: 標準的なシリアライズファイルデータ

```
T01:000005
T02:001006
T03:99
T04:
:0300000000096B89
:03000300000005F5
:02000C005A0197
:01003F004F71
:00000001FF
```

上記ファイルデータの内容は
ライン T01:ー 現在のシリアルナンバー・0x000005
ライン T02:ー 次回のシリアルナンバー・0x001006
(この番号は次回のシリアル番号書込み時に使用する番号)
ライン T03:ー T04で使用するデータフォーマット(INTEL. hex)
ライン T04:ー デバイスに書き込む前にプログラマのバッファにロードされるデータ
0x0番地から0x00, 0x09, 0x6B
0x3番地から0x00, 0x00, 0x05
0xC番地から0x5A, 0x01
0x3F番地に0x4F

オプション レコード

T05:< message > 警告またはエラーメッセージ。
このレコードは、シリアライズが中止になった時、PG4UW のソフト上に警告またはエラーメッセージとして表示する < message > です。

T06:現在のシリアルナンバーがリミットをオーバーしたことを示す。
このレコードは、Eコマンドラインパラメータを指定している際に、使用される。ダイアログ上のシリアライズの最終シリアル番号がゼロでないことを意味する。

T11:< message > 動作情報等のメッセージ。情報を表示後プログラマは動作を続ける。

Custom generator モードを使ってデバイスに書き込む場合の動作フロー

Custom generator モードが使われるとき、シリアル番号用.dat ファイルを作成するために シリアル番号生成プログラムが実行されます。
コントロールソフト PG4UW はシリアル番号生成プログラムから出されたコマンドラインパラメータをチェックします。
.dat ファイルに作られたシリアル番号データは、プログラマのバッファにすぐに読まれて、デバイスに書き込む為のデータとして使われます。
また、次回に使用するシリアル番号情報（T02 レコード）は、PG4UW に記憶されます。

一般的なデバイス書き込み時の動作フローを以下に記載します:

1. 書き込み作業をスタートさせます。
2. テスト用デバイスをプログラマのソケットに実装します。
3. シリアライズは4つのステップで構成されています。
   －シリアル番号用.datファイルを作成するために、コマンドラインパラメータを使用して シリアル番号生成プログラムが呼び出されます。
   －シリアル番号用ファイル.dat が利用可能になるまで待ちます。
   －シリアル番号用.dat ファイルからプログラマのバッファにシリアル番号をリードします。
   －読込んだシリアル番号は.dat ファイルから削除します。
4. デバイスの書き込みを行ないます。
5. デバイスのベリファイを行ないます。
6. 書き込み結果をチェックします。

   PASS/FAIL チェックはコントロールソフト PG4UW によって完全に制御されています。
   シリアル番号生成ソフトはいかなる場合でも、プログラマの PASS/FAIL 結果に影響を及ぼすことはありません。PG4UW はシリアル番号生成ソフトからコマンドラインパラメータを与えられて、呼び出されるだけです。
   書き込み OK 時　PG4UW が次のシリアル番号を要求します。シリアル番号はステップ３で.dat からリードされます。
   シリアル番号生成ソフトを CALL する時に、コマンドラインパラメータで次の番号を指定しておきます。
   書き込み ERROR 時　PG4UW は新しいシリアル番号を要求しません。
   このシリアル番号は次のデバイスで使用されます。
   シリアル番号生成ソフトを CALL するときに、このシリアル番号をコマンドラインパラメータで 指定しておきます。
7. 次のデバイスに書き込む
   Yes － ステップ2に進む
   No － ステップ8に進む
8. 書き込み作業を終了する。

Note：書き込みエラーが発生した場合でも、この時のシリアル番号は使用されます。
書き込みに失敗した時に使用した番号が、次のデバイスに書き込まれます。
シリアル番号用.dat ファイルにエラーが検出された場合、PG4UW はエラーの発生をユーザーに知らせ 書き込み動作をすぐに停止します。

デバイス->デバイスオプション->カウント＆カウントダウン

カウントはプログラマで処理したデバイスの数量を数える機能です。

Program、Verify、Blank、Erase、Readチェック・ボックス

PASS 数をカウントするプログラマの動作モードを指定します。

デバイスの書込みや Verify を実行した場合、最終結果として PASS または FAIL カウンターを＋1増やします。同時に合計カウンターを＋1増やします。
また実際のプログラム動作等では Program＋Verify のように複数の動作を実行しますが、カウンターは＋1だけ増やします。

例：Program では Program＋Verify 動作します。
Erase では Erase＋Blank 動作します。
両方とも PASS または FAIL カウンターを＋1だけ増やします。

カウントダウン チェック・ボックス

カウントダウンの Enable 又は、 Disable を設定します。カウントダウンに続くエディット・ボックスにデバイス数量（最初に設定する目標数量）を指定します。

カウントダウンはユーザーがこれから処理するデバイスの数量をあらかじめ設定しておきます。
デバイス書き込みが正常終了した時に、この数値を－1（カウントダウン）します。
ユーザーの予定数量が終了（カウント＝0）したとき、作業完了メッセージを表示し、カウントダウンモードを Disable にします。

カウントダイアログはメイン画面上のカウント部分をマウス右ボタンでクリックすることでもオープンできます。作業中のカウント数はメイン画面上のカウントパネルに表示されます。

カウント数

カウントダイアログには 8 個の項目があります。

PASS ：正常終了したデバイスの数
FAIL ：指定モードで不良になったデバイスの数
変換アダプタ ERROR ：モード開始前に実行したアダプタのテストエラー数
挿入テスト ERROR ：モード開始前に実行したデバイスチェックエラー数
ID チェック ERROR ：モード開始前に実行したデバイスの ID チェックエラー数
動作キャンセル ：ユーザーによるキャンセルされた数
その他の FAIL ：上記以外でエラーになった時の数
合 計 ： 全ての合計

カウント表示部分

表示パネルには 4 個の数量カウンターとカウントダウン情報を表示します。

PASS ：正常終了したデバイスの数
FAIL ：指定モードで不良になったデバイスの数
他の FAIL：上記以外でエラーになった時の数
合 計 ： 全ての合計

カウントダウン： Enabel 又は、 Disable を表示
残数 ：作業完了までの残りのデバイス数と最初に設定した目標個数

操作ボタン

カウントのリセット :カウント値をリセットします。
再設定 :カウントダウンに初期値を再ロードします。

Note:
新しいデバイス・タイプが選択されたとき、すべてのカウント値はゼロにリセットされ、そして、カウントダウンはDisable(解除)されます。
またPG4UWソフトを終了した時に、上記のカウント＆カウントダウン情報は LOG ファイルに記録されます。

デバイス->デバイスオプション->関連ファイル

デバイス->デバイス選択／履歴を使用してデバイスを変更したとき又は、コントロールソフトを起動した後にユーザーが使用するデータファイルを自動的に読込むことが出来ます。

・起動時自動読込み:Yes(自動読込み)又は、No(読込み無し)を設定します。

・ファイル名:読み込むデータファイル名を指定します。
ファイル名には絶対パス名(フルパス)を付けて下さい。

ここで設定した関連ファイル情報は ファイル->保存して終了 コマンドでDISKにセーブされます。

デバイス->デバイスオプション->Special

選択しているデバイス専用の機能をサポートするためのものです。
そのため、選択したデバイス毎に表示される機能が異なります。

例: FLASH PROM が選択されている時
メニューは **デバイス->デバイスオプション->セクター表示/編集**
になります。このデバイスのコンフィグレーション情報を表示したり、変更したりすることが出来ます。

デバイス->Blank　check

デバイスの未書き込み状態をチェックします。
動作経過やチェック結果をINFOウィンドウに表示します。
デバイスオプション　->　動作オプション　で動作範囲を変更することが出来ます。

デバイス->Read（Copy）

デバイスに書き込んであるデータをパソコン上のバッファメモリにリードします。
拡張メモリやコンフィグレーション情報を持ったデバイスでは、これらのデータもリードする事が出来ます。また一部のデバイスではコンフィグレーション情報を変更することも出来ます。
動作経過やチェック結果をINFOウィンドウに表示します。

デバイス->デバイスオプション->動作オプション内の"Read後にVerify"を設定すれば、
デバイスのデータをリード後にVerifyを実行しますので、マスターデータの信頼性をチェックすることが出来ます。

デバイス->Verify

パソコン内のバッファメモリデータとターゲットデバイスに書かれたデータを比較します。
動作経過やチェック結果をINFOウィンドウに表示します。
デバイスオプション　->　動作オプション　で動作範囲を変更することが出来ます。

オプション->オプション設定->ベリファイエラーファイルを設定することにより、エラー情報をVERIFY．errファイルに記録することが出来ます。
Log・WindowにはVerifyエラーが発生した時点から最大45個分のエラー情報を表示します。

注意；

1. Verify 動作はデバイスに書かれたデータを毎回リードして、プログラマ内のバッファメモリと比較しています。
   不安定な書き込みデバイスでは、リードする時の周囲温度やソケットの接触具合によりエラーが発生する場合があります。
   デバイスを消去してから再プログラムする事を推奨します。

2. プロテクト機能を持ったデバイスでは、一旦リードプロテクトを行うと正しいデータをリードする事が出来ません。そのため Verify エラーになります。

デバイス->Program

パソコン内のバッファメモリデータをターゲットデバイスに書きこみます。
動作経過やチェック結果をINFOウィンドウに表示します。

デバイス->デバイスオプション->動作オプションを使用してデバイスの書き込み範囲を変更することが出来ます。
またデバイスによっては各デバイス固有の動作設定等を指定することも出来ます。

デバイス->Erase

電気的な信号で消去可能なデバイスを指定した場合に操作可能になるモードです。
ターゲットデバイスに書かれているデータを消去します。
動作経過やチェック結果をINFOウィンドウに表示します。

消去後、デバイスデータが完全に消去できたか確認する為に、Blank チェックを実行しています。
但しBlankチェック以外の方法で消去確認ができるデバイスでは、こちらの方法で消去の確認を行っています。

デバイス->Test

サポートデバイスから Static RAM(SRAM)を選択した時、有効になる機能です。

SRAM テストは下記の 3 種類のチェックが行われます。

1) デバイスの出力ピン機能テスト
出力信号テスト:D0..D7 ピンを CE/, OE/, WE/ 信号を使用してチェックします。
- 最初に SRAM アドレス 0x0 に データ 0x55 をライトします(CE/=L WE/=LOE/=H)。
その後 このアドレスからデータをリードし、比較チェックします(CE/=L WE/=HOE/=L)。
- その後信号の組み合わせを変更して(CE/=L WE/=H OE/=H), (CE/=H WE/=H OE/=L), ..., SRAM からデータがリード出来ない事もチェックしています。

2) SRAM テスト, 標準的なチェック
プログラマは SRAM デバイスにランダムデータをライトします。
その後ライトしたデータを Verify して SRAM をチェックしています。

3) SRAM テスト, オプションテスト
オプションテストとして、SRAM の"Walking one" (テスト時に"1"をライトする) と"Walking zero" (テスト時に"0"をライトする) テストを指定することが出来ます。

Note:
- ライト動作後のリードまでのディレイ時間を選択することが可能です。
ディレイ時間の選択によって、デバイスの時間によるデータ変化(主にリーク不良)を検出することが出来ます。
- プログラマはデバイスの特定ピンの過大電流や、アナログ的な信号はテスト出来ません。
- 全てのテストは SRAM 動作スピードよりも遅いスピードでテストされています。(プログラマの最高 Verify スピードではテストしています)
そのためプログラマによるテストは SRAM の動作を保証するものではありません。

デバイス->IC test

IC test は、主に汎用 Logic IC をチェックする機能です。テストする IC はデバイスの種類によって分類されています。

最初にテストするデバイスの種類を選択し、その後テストするデバイス名とテストモード(LOOP,SINGLE,STEP)を選択します。
テスト方法はソケット上の IC にプログラマ側から"H"、"L"パターン(テストベクター)を加え、デバイスが出力する信号及び結果を Log Window 上に表示します。
ユーザーが定義したテストベクターを使用してテストすることも出来ます。テストベクターの作成方法は 'example_e.lib file' を参照して下さい。

注意:
・IC test はテストベクターを使用して遅い速度(プログラマが発生するパターン速度)で動作します。
またテストに使用しているテストベクターは IC の一部の機能しかチェック出来ません。
テスト結果が "FAIL" の場合はテスト IC に何らかの欠陥があります。
テスト結果が "PASS" の場合はこの IC test で "PASS" になりましたが、デバイスのアクセス時間に関するテストは行っていませんので注意して下さい。
・プログラマ側から出力する信号の立ち上がり／立下り波形はプログラマブル・デバイス用に調整されています。
急峻な立ち上がり／立下りが必要な一部のデバイス(カウンター IC)では正常に動作しない場合があります。

デバイス->Jam/VME/SVF/. . . Player

Jam STAPLは、Altera社によって開発され、Logicデバイスメーカー(PLDメーカー)、書き込み装置メーカー、テスト装置メーカーによってサポートされました。
Jam™の標準言語STAPLとJEDEC標準JESD－71は、ISP(インシステム・プログラミング)用の標準フォーマットです。JamSTAPLはオープンライセンスで広く世界で使用することが出来ます。
JTAG(IEEE 1149.1)を使用してプログラマブル・デバイスの書き込みや電気回路のテスティングをサポート出来ます。
JamSTAPLは書き込みや確認は出来ますが、デバイスからデータをリードする機能は認められていません。

Jam STAPL書き込みには2つの機能があります。Jam ComposerとJam Playerです。
Jam Composerはプログラムで、一般的にはプログラマブルLogicデバイスメーカーがJamファイルを作成します。このJamファイルには、設計したLogic回路を書きこむためのアルゴリズムとユーザーデータが含まれます。
Jam PlayerはJTAGチェーンを使用してデバイスに書き込みやテストを行なうために、Jamファイルを読み込み、テストベクターを出力します。
デバイスは、プログラマのZIFソケットもしくは、ISPコネクタを使用してユーザーデータが書き込まれます。
このJamファイルが使用可能なデバイスはコントロールソフト上のデバイス選択時にデバイス名の後に[PLCC44](Jam)か(ISP-Jam)が表示されています。
複数のデバイスをJTAGチェーン接続して、書き込み、テストが可能です。

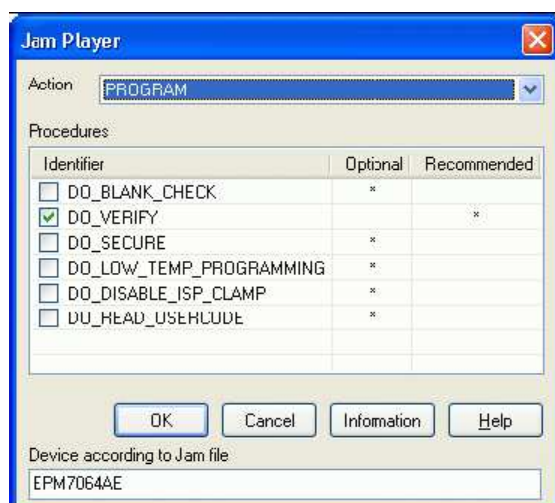
より詳しい情報は、下記ホームページにあります。

http://www.altera.com

アプリケーション・ノートを参照してください。
“AN 425: Altera Devicesを書くためのJamPlayerの使用方法”
“AN 100: In-System 書込みを使用時のガイドライン”
”AN 122: ISP書込みのための Jam STAPL の使用方法“

Software tools:

Altera: MAX+plus II, Quartus II,
SVF2Jam utility (converts a serial vector file to a Jam file),
LAT2Jam utility (converts an ispLSI3256A JEDEC file to a Jam file);
Xilinx: Xilinx ISE Webpack or Foundation software
(Generates STAPL file or SVF file for use by utility SVF2Jam);
Actel:Actel Libero® Integrated Design Environment (IDE) (generates STAPLE file and/or PDB file),Actel FlashPro (converts a PDB file to STAPLE file);

JAM player dialog

JAM Player version2

Jam Playerボタン、ダイアログ等の関連メニューはJTAG対応デバイスを選択しないと表示されません。たとえば"Xilinx製 XC2C32A"を選択する必要があります。

Action

起動したい動作モードを選択してください。
Jam file Ver2は各種の動作モードで構成されています。各動作モードにはデバイス書き込みに必要な手順が入っています。

Procedures(Player Ver2)

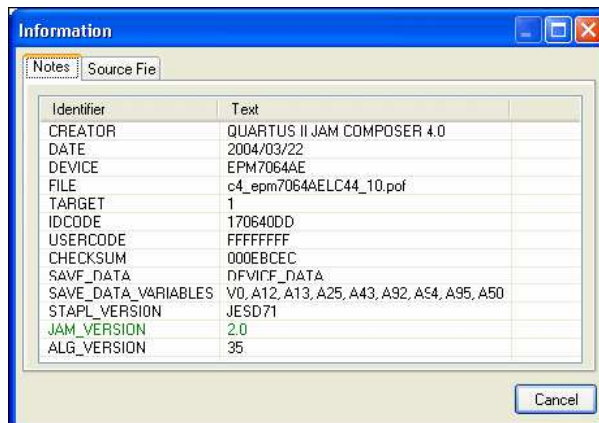
書き込みフローは、それぞれの手順から、命令を実行します。
手順は、オプション選択か、推奨選択を使用します。推奨された手順にはあらかじめチェック印がつけられています。必要に応じてこの設定をEnable、Disableに変更出来ます。
Jam Playerは、チェック印のつけられた手順だけ実行して、その他は無視します。
選択の数は、それぞれ異なり、Jam fileに依存します。

OK :マーキングした**Action**を確認後 OK ボタンを押します。

Information :Jam file情報を表示します。
またソースファイルや注意書きを見ることが出来ます。

Jamファイル内で指定したデバイス名

Jam fileは特定したデバイス専用に作られています。使用するデバイス名はJam file内で指定します。この名前とプログラマで選択したデバイス名が同じでなければいけません。
もしこの名前が違うときはJam player動作時にエラーメッセージを表示します。

JAM FILE information dialog

Note :

Note タグは Jam file についての情報を表示します。
この Note 内の情報は、選択したデバイス用の Jam Program に関連したドキュメントとアトリビュート(属性)を含んでいます。

Source file :

Source タグは Jam 言語で書かれたプログラムです。
Jam プログラムは、"Statement" の順序で構成されています。
Jam "Statement" はラベル、命令、引数、終了記号のセミコロンによっ構成されています。
引数には、リテラル定数、変数、要求されたデータ方式(関数もしくは整数)が入ります。
それぞれの"Statement"は、Jam プログラム内では 1 行で書かれますが、数行の場合もあります。
Jam プログラムではコメント行を除いて、 1 行以上になることに問題はありません。 アポストロフィ(')はコメントマークで、インタープリタ(プログラマに組み込まれている翻訳ソフト)では無視します。
Jam 言語は作成されたプログラムの大きさ、行数、また "Statement" の文字数に制限はありません。

Jam file の拡張子.jbc は Jam STAPL と呼ばれるファイルで一般的な Text エディターでは表示できません。

XILINX デバイス用JED File を Jam STAPL File に変換する

JEDファイルをJam STAPLファイルに変換する方法

1.Xilinx 製フリーソフトをインストールします。

Environment (ISE) 6.3i software free download: WebPACK_63_fcfull_i.exe + 6_3_02i_pc.exe (315MB or so)

2. ソフトを起動します。起動ソフト名：Xilinx ISE 6/Accessories/iMPACT

- ダイアログ "Operation Mod Selection: What do you want to do first?" 内の "Prepare Configuration Files" を選択。
- ダイアログ "Prepare Configuration Files: I want create a:" 内の "Boundary-Scan File" を選択。
- ダイアログ "Prepare Boundary-Scan File: I want create a:" 内の "STAPL File" を選択。
- ダイアログ "Create a New STAPL File" 拡張子を.Stapl にした Jam ファイル名を入力する。
- ダイアログ "Add Device" で拡張子が .jed なっている JED ファイルを選択する。
- JTAG チェーン内でデバイスを選択し、動作項目を選択する。(e. g.: Erase, Blank, Program, Verify; right mouse button)
- メニュー内の項目 "Output/Stapl file/Stop writing to Stapl file" を選択する。

3. プログラマ用コントロールソフト PG4UW を立ち上げ

Jam対応デバイスを選択し(e.g.:Xilinx XC2x32A[QFG32](Jam))

Jamファイルをロードする(File type :STAPL File)。

4. コントロールソフトの デバイス->デバイスオプション->動作オプション 内の"Jam configuration" ボタンをクリック Warning"Select device from. . . . " ' "Yes"を選択。

5.ダイアログ JAM Playerで動作させる項目を選択し、最後にToolバー部分の "Play Jam" ボタンを押して実行する。

STAPLE ファイルを使用してデバイスをプログラムする

Actel 社製 Flash FPGA デバイスをプログラマでリード／ライトする為には、Actel 社の Jam player を使用して作業します。

Jamプログラム対応デバイスを選択した場合、プログラム画面上のツールバー(Program, Ersae, Verify...)ボタンは "STAPL" ボタンに変更されます。

Actel デバイスの操作 (program, erase, verify...) には以下のいくつかのステップがあります。

・STAPLE ファイル(*.stp)をロードする

メインツールバー上の"Load"ボタンをクリックして、使用する STAPL ファイル (例えば Actel 社の回路設計ソフト LIBERO IDE で作成したファイル)を読込みます。
このファイルにはユーザーが作成した回路データとデバイスをプログラムするためのアルゴリズムが組み込まれています。

・動作モードを選択する

STAPLE ファイルのロードが完了したら、デバイスの動作オプション内から動作させるモード (Alt+O shortkey)STAPL configuration. を選択して下さい。
デバイスをプログラムするためには動作リスト内の 'PROGRAM' を選択します。

デバイス操作に関するこれらの動作説明は ACTEL 社 'FlashPro User's Guide' <http://www.actel.com>を参照して下さい。

・選択したモードを実行する

選択した動作を実行させるために 'STAPL' ボタンをクリックします。
動作が正常終了すると LogWindow 画面上に"Exit Code =0... Success"を表示します。

*.PDB ファイルを JAM STAPLE ファイルに変換する

***.PDB** ファイルは Actel 専用プログラマで使用するファイルフォーマットです。
プログラマでは Actel 社製デバイスは Jam STAPL ファイルのみに対応しています。そのため ***.PDB** ファイルから Jam STAPL ファイルへ変換する必要があります。

***.PDB ファイルを Actel デバイス用 Jam STAPL ファイルに変換する方法**

1. 'FlashPro' ソフトをインストールして下さい。
   このソフトは Actel 社の開発ツール 'Actel Libero' に含まれています。または <http://www.actel.com> サイトから標準バージョンソフトをダウンロード出来ます。
2. 'FlashPro' を起動させます。
   プログラマの 'プロジェクトを保存' ボタンをクリックして新しいプロジェクト名を入力して下さい。
   動作させたいモードを選択して、'Single device' または 'Chain device' を選択して 'OK' ボタンをクリックして下さい。
   Configuration メニュー内の 'ロードするファイル選択 --> ***.PDB** ファイル変換を選択する。
   外部に作成するファイル 'SingleDeviceSTAPL ファイル...' を選択して、ファイル名を入力し、保存ボタンをクリックして STAPL ファイルを作成して下さい。

3. ***.PDB** ファイルが STAPL ファイルに変換されます。
   Actel デバイスのプログラム で使用可能な ***.STP** ファイルが作成されます。

Actelデバイスでよくある質問

Q： 既に書込み済みの Actel デバイスはどのようにして ID チェックと Verify を行うのですか？

A： ID チェックや Verify を行うためのオプションがあります。
各々のオプションは既にプログラムされた Actel デバイスとロードした STAPL ファイルを比較する方法で行います。
以下に STAPL ファイルを使用した場合の比較方法を記載します。

・DEVICE_INFO： デバイスをリードして、このデバイスに書かれているデータのチェックサムを LogWindow に表示して下さい。
このチェックサム値はプログラマのマニュアル操作で比較することが出来ます。
STAPL ファイル側のヘッダー情報は Information Window に表示されますので、この値とデバイスをリードした時の値を比較します。

・VERIFY_DEVICE_INFO： 前の機能と類似したオプションです。デバイスのチェックサムと STAPL ファイルのチェックサムを自動で比較します。 比較した結果はメッセージ Window に"SUCCSES" または "ERROR" 表示されます。

・VERIFY： プログラムされたデバイスの内容と STAPL ファイルの内容を比較します。
最も安全ですが、動作スピードは最も遅くなります。デバイスの容量に依存しますが数十秒以上かかります。
これに比べ前記のチェックは数秒で終了します。
選択された比較機能（FPGA アレイ, ターゲットの FlashROM, セキュリティ設定...)は bit 単位でベリファイされます。

Q： Actel デバイスに 2 個の STAPL ファイルを一度の操作でプログラムすることが出来ますか？

A： 可能です。プログラマのマルチ・プロジェクト機能が利用出来ます。
例えば最初のSTAPLファイルに書込み用のデータを入れ、2個目のSTAPLファイルにセキュリティ用のデータを入れて、作業をすることが出来ます。

VME の概要

The ispVM Virtual Machine は IEEE1149.1 標準バンダリースキャンテストと互換性のある特定のデバイス用に作られた仮想マシンです。
The ispVM EMBEDDED ツールはラティス社の"ispVM Virtual MachineTM" をサポートしていて、バンダリースキャン書き込みとテスト用に工業標準規格のシリアルベクタフォーマット（ SVF ）言語が使用できます。
ispVM ソフトは VME ファイル（ ispJTAG ファイルと Lattice 以外の IEEE1149.1 の標準 JTAG ファイル ）を作成します。
VME ファイルは、ispVM システムウインドウからの情報を見ることができる Hex コードのファイルです。
デバイスは、プログラマの ZIF ソケットもしくは、ISP コネクタを使用してユーザーデータが書き込まれます。
この VME ファイルが使用可能なデバイスはコントロールソフト上のデバイス選択時にデバイス名の後に [PLCC44](VME) か（ ISP-VME ）が表示されています。

より詳しい情報は、下記ホームページにあります。

http://www.latticesemi.com

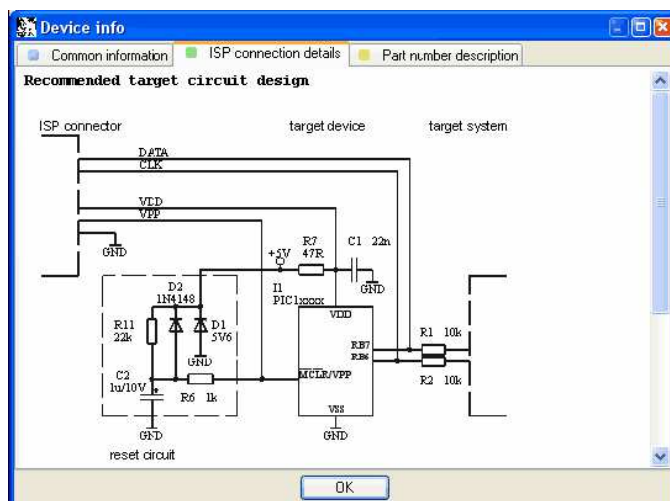
Software tools:

Lattice: ispLEVER, ispVM System ISP Programming Software,PAC-Designer Software, Svf2vme utility (converts a serial vector file to a VME file)

デバイス->デバイス情報

現在選択しているデバイスの詳細情報を表示します。

デバイスのデータ構成、容量、書き込みアルゴリズム、対応変換アダプタ等を表示します。
それ以外にデバイスのパッケージ寸法、デバイス型番の詳細、およびISPデバイス選択時には回路設計情報等も表示します。

またショートカット＜Ctrl＋F1＞keyでこのデバイス情報を表示します。

例：ISPコネクタとデバイスの配線接続例やターゲットデバイス回路例を表示

プログラマ（メインメニュー・コマンド）

プログラマ関連の操作を集めたメニューです。
プログラマとパソコンの接続、Automatic・Yesモード（ソケット上のデバイス交換を自動判断して書き込みを自動的にスタートする機能）、プログラマの自己診断が入っています。

プログラマ->プログラマ検出

パソコンとプログラマを新規に接続する場合に使用します。
このダイアログには次の項目があります。

Programmer　　:新しく接続したプログラマの機種を指定します。

Port　　:プログラマとパソコンを接続するポートを指定します。

USBポートに比べLPTポート（プリンターポート）接続は動作が大幅に遅くなります。
SpecialLPTポートはLPTポートアドレスを特別に割り当てている場合に指定します。

＜Enter＞keyまたは“接続”ボタンを押すとプログラマの検出が始まります。
この動作はコントロールソフトを起動したときと同じです。

プログラマ->プログラマ再検出

プログラマとパソコンを再接続する場合に使用します。再接続ですので、プログラマ機種の再指定はありません。

接続ケーブルを変更する、接続USBポート番号を変更する、使用するプログラマを変更する場合は前記の プログラマ->プログラマ検出 を使用してください。

プログラマ->Handler

Reserve（予約コマンドです。使用できません）

プログラマ->Module　オプション

Reserve（予約コマンドです。使用できません）
ギャングプログラマで使用するソケット番号と使用しないソケット番号を指定するコマンドです。
このプログラマでは使用出来ません。

プログラマ->Automatic　YES　!(自動実行モード)

自動書込みモードの設定に使用します。このモードはオペレータがソケット上のデバイスを取り去り、新しいデバイスをソケットにのせると、自動的に書き込みをスタートさせる機能です。
プログラマは新しいデバイスがソケットに挿入されたことを自動的に検出し、スタート操作なしに動作を実行します。
ZIFソケット上のデバイス交換時に＜ESC＞keyを押すと作業を中止出来ます。

・デバイスの書き込みが終了すると、
GOODまたは**ERROR**　LEDが点灯し、**BUSY**　LEDを点滅させます。
・デバイスをソケットから取り去ると、
GOODと**ERROR**　LEDを消し、**BUSY**　LEDを点灯させます。
この状態はプログラマの書込みが終了し、次の新しいデバイスを待っていることを示します。
・デバイスが正しく挿入されると
BUSY　LEDを点滅させ、書込みを開始します。
・デバイスの誤挿入が検出されるか、待ち時間が過ぎると
ERROR　LEDを点灯させます。
（待ち時間＝デバイス挿入完了までの設定時間）

Automatic・Yes　：モードのEnable／Disableを設定します。
プログラマ->プログラマ検出で新しいプログラマを設定した場合、このモードはDisable設定になります。
注意　：このモードは ISP プログラムでは利用出来ません。

動作開始反応時間　：デバイスをソケットに挿入してから書き込みを開始するまでの時間です。
デバイス挿入後に位置を修正する時間が必要な場合は長い時間の“ウェイト有り”設定にします。

使用する変換アダプタ　：使用する変換アダプタ名を表示しています。

チェックから除外した ZIF ソケットピン番号　：ピンチェックを SKIP するピン番号（ZIF 上の番号）を表示しています。

Automatic・Yes！チェックピン再設定　：設定済みのパラメータでAutomaticYES　が動作しない場合に、ピンチェックを SKIP するピン番号を再設定します。
再設定用の Window がオープンしますので、メッセージに従いピンチェック作業を行って下さい。

デバイスを取り去るまでの時間　：書き込み終了したデバイスをZIFソケットから取り去ってから、次の新しいデバイスのピンチェック開始までの時間を指定します。
１から１２０秒が設定出来ます（ディフォルトは2秒）

デバイス挿入完了までの時間　：デバイスチェックを開始してから、ERROR＿LEDが点灯するまでの時間を設定します。
１から１２０秒が設定出来ます（ディフォルトは5秒）

エラー発生時に一時的に中断する ：モードのEnable / Disable を設定します。

プログラマの動作中にエラーが発生した場合に、Automatic YES動作を一時的に中断する**"Enable"**か、動作を継続する**"Disable"**かを設定します。

この設定情報は**オプション–>オプション設定を保存** コマンドでDISKに保存されます。
また "ファイル／プロジェクトを保存" 操作でも保存されます。

注意：Automatic YES！モードはZIFソケットとデバイスPINの接触を検出して動作します。
アダプタ上に部品が搭載されている場合、Automatic YES！モードモードが正常に機能しない場合があります。この場合はマニュアル操作で作業をして下さい。

プログラマ->セルフテスト

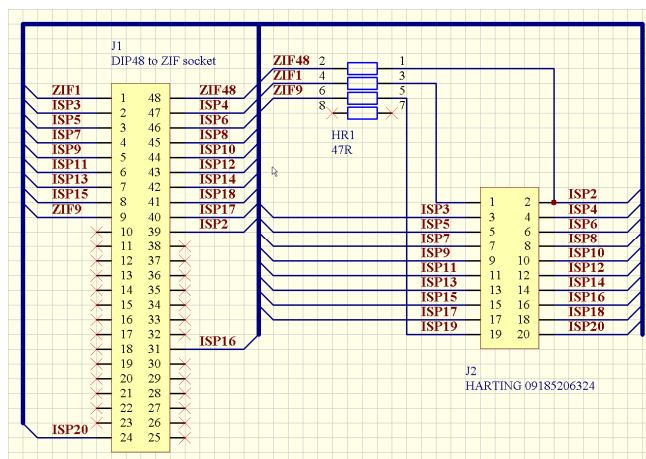
プログラマの自己診断を行なうコマンドです。
この自己診断ではプログラマの内部回路をチェックします。チェック用の治具（ Diagnostic POD ）は必要ありません。
プログラマに必要な機能を全てチェックするためには、次のセルフテスト プラスを実行してください。

プログラマ->セルフテスト　プラス

チェック用の治具（Diagnostic　POD）を使用して、プログラマの自己診断を行なうコマンドです。
この自己診断ではプログラマの内部回路とZIFソケットに出力する信号もチェックしています。

Note：Diagnostic　PODはプログラマ購入時に標準品として添付されています。
この自己診断を1ヶ月に1回実施されることを推奨します。このテストでデバイス書き込みに必要な信号の確認をしていますが、プログラマの信頼性を維持するためにメーカーによる定期的（1年毎）な校正、点検を推奨します。

プログラマ->セルフテスト　ISPコネクタ

チェック用の治具（ISP　Diagnostic　POD）を使用して、プログラマの自己診断を行なうコマンドです。この自己診断ではプログラマの内部回路とZIF（48ピン）ソケットおよびISP(20ピン)コネクタに出力する信号をチェックしています。

（1）チェック方法はプログラマのZIFソケットに“ISP　Diagnostic　POD”を取り付けます。
（2）プログラマに付属の20ピンフラットケーブルの一方をこのPOD上の角型コネクタに接続します。
（3）ケーブルの反対側をプログラマ手前のISPコネクタ（20ピン角型コネクタ）に接続します。
（4）接続完了後、プログラマの プログラマ->セルフテスト　ISPコネクタ コマンドを実行します。

Note：ISP　Diagnostic　PODと20ピンISPケーブルはプログラマ購入時に標準品として添付されています。
この自己診断を6ヶ月に1回実施されることを推奨します。このテストでISP書き込みに必要な信号の確認をしていますが、プログラマの信頼性を維持するためにメーカーによる定期的（1年毎）な校正、点検を推奨します。

プログラマ->キャリブレーションテスト

Reserve（予約コマンドです。使用できません）
メーカーにてプログラマをメンテナンスするためのコマンドです。

オプション（メインメニュー・コマンド）

プログラマの基本的な設定を行なうコマンドを集めたメニューです。

オプション–> オプション設定

プログラマの基本的（ファイル設定、言語、ログ・ファイル、設定の保存）設定を行ないます。
このオプション設定はPG4UWソフトを終了した時に自動的に保存されます。
また オプション –> オプション設定を保存 でいつでも保存する事が出来ます。

オプション–> オプション設定–>Load ファイル オプション

ファイルを読み込むためのファイルフォーマット選択、ファイルの再読み込み、バッファデータの消去を指定するオプションです。

読込み前にバッファデータ消去

Yesチェック :ファイルを読み込む前にバッファメモリをValueデータで初期化します。
Noチェック :ファイルはバッファメモリに上書きされます。

現在のファイルが他のプロセスで変更された場合の処理

使用中のデータファイルが他のアプリケーションによって書き換えらときの再読込み処理を指定します。

読込み確認 :データに変更があった場合、再読込みする前にメッセージを表示する。
自動–再読込 :データに変更があった場合、自動的に再読込みする。
無　視 :データに変更があっても、再読込みしない。

データ変更は次の場合に確認、チェックされます。
・他のアプリケーションからコントロールソフトに切替わったとき。
・デバイス動作でVerifyまたはProgramが選択されたとき。
・デバイス動作で“Repeat”が選択されたとき。

読込時のファイルフォーマット:データファイルを読み込む時のファイルフォーマットの判別方法を指定します。

自　動 :プログラマが判別可能なファイルフォーマットを自動で認識します。
自動判断したデータフォーマットでバッファメモリに読み込みます。
マニュアル選択 :ファイル読込ウィンドウでフォーマットを指定します。
読み込むファイルと指定したフォーマットが異なる場合は、データは正しく読み込めません。

プログラム起動時にプロジェクトリストを表示する

チェック有り時 :プログラマ起動時にプロジェクトリストを表示します。

オプション-> オプション設定->ファイル 拡張子

読み込むファイルフォーマットの拡張子とプロジェクトファイルの拡張子を指定します。

ファイル フォーマット マスク ：データファイルを **ファイル読込** と **ファイル保存** するときに、表示するファイルの拡張子でフィルターをかけて表示されるファイル数を少なくします。マスク設定にワイルドカード（＊、？）が必要です。

Note：一個のフォーマットマスクに複数の拡張子を指定することも出来ます。
Exsampl：Motorola　＊.MOT,＊.S19
このように 2 個記述した場合 ＊.MOT と ＊.S19 の拡張子が付いたファイルが Motorola ファイルとして抽出されます。

プロジェクトファイルの拡張子（ディフォルト）：プロジェクトファイルの拡張子を指定します。

オプション-> オプション設定->デバイス選択時のバッファ

デバイス選択時にバッファメモリ内容をクリアしたい時に使用します。

デバイス選択時にバッファメモリ内容を消去する

このオプション設定により、デバイスを選択した時にバッファメモリ内容を自動で指定データにクリアすることが出来ます。

消去時データ

- 自動設定 を選択した時は使用するデバイスの BLANK 状態にクリアします。
- ユーザーが指定する を選択した時はその後ろにある - 消去データを指定する部分に入力したデータでクリアします。

指定したバッファファイルのパスを有効にする

このオプションは特定のバッファエリアをファイルデータでクリアする時に指定します。
この機能は動作が遅いためあまりお勧めはしません。

オプション-> オプション設定->言　語

メニュー、ボタン、ダイアログ、情報、メッセージ等で使用する言語を選択します。
ヘルプ画面には別の言語を選択することも出来ます。

オプション-> オプション設定->サウンド

コントロールソフトが使用するサウンドモードを指定します。 コントロールソフトはデバイス動作終了時等に音を鳴らします。
また警告やエラーメッセージを表示した時にも音を鳴らします。また音なしの設定も出来ます。
ユーザーはパソコンに用意されている Windows システム上の音をこのメニューで指定出来ます。

次のアクションの後に音を鳴らす：

操作が正常に行われた時のチェック BOX：

- チェック有り：デバイスの PROGRAM や VERIFY 等が正常終了した時に音を鳴らします。
- チェック無し：音は鳴らしません。

エラー発生時のチェック BOX：

- チェック有り：デバイスの PROGRAM や VERIFY 等が ERROR 終了した時に音を鳴らします。
- チェック無し：音は鳴らしません。

プログラマ内蔵スピーカのサウンド設定：

一部のプログラマはスピーカを内蔵しています。このプログラマを接続した時に選択可能になります。デバイス動作終了時のサウンド指定です。

オプション–> オプション設定–>ベリファイエラーファイル

このオプションを設定した場合、Verifyエラー時のデバイスアドレス、デバイスデータおよびバッファデータをファイルに記録します。またLog・WindowにはVerifyエラーを最大45個まで表示します。

エラー設定は

ベリファイエラー情報をファイルに保存：

・No　チェック　　：PASS／FAIL結果をLog・Windowに表示する。

・新規保存　チェック　　：エラー情報をLog・Windowに表示すると同時に、Verifyエラーファイルに保存します（以前のデータは削除されます）。

・追加保存　チェック　　：エラー情報をLog・Windowに表示すると同時に、Verifyエラーファイルに追加します。

エラーファイル名：　　：記録するファイル名を指定する。
ディフォルトのファイル名は“Verify. err”ですが、変更する場合はここで指定します。

エラー情報ファイルの制限：

1回のベリファイ動作で保存する最大エラー数を指定します。

－ チェック有り：　指定した最大エラー数でファイルの記録をストップします。
－ チェック無し：　全てのエラーアドレス、データをファイルに記録します。

オプション–> オプション設定–>ログ・ファイル

ログ・ファイル設定を行なうと、Log・Window上に表示した情報をログ・ファイル内にも記録します。
使用されるログ・ファイルネームは、“Report. rep”です。
コントロールソフトは、ログ・ファイル名ボックスに入力したファイル名とディレクトリにこのログ・ファイルを作成します。

ログ・ファイル:

No(ディフォルト):ログ・ファイルを作成しません。プログラマの動作情報は、Log・Windowに表示します。

新規保存　:コントロールソフトを起動するたびに、古いログ・ファイルを削除して新しいログ・ファイルを作ります。

追加保存　:Log・Window上の情報を過去に使用していたログ・ファイルに追加して保存します。ログ・ファイルが存在しない場合、新しいファイルが作成されます。

ログ・ファイル名:ログ・ファイルを記録するファイル名を指定します。

ログ・ファイル名に日付を追加する
:ログ・ファイルを記録した年月日をファイル名に追加します。

例:ユーザーが指定したファイル名が　c:¥logs¥myfile. log
最後にログ・ファイルを使用した年月日が2007年5月30日の場合
作成されるログ・ファイル名は
c:¥logs¥myfile-2007-05-30. log

例:ファイル名を年月日だけにするには
ユーザーが指定するファイル名を　c:¥logs¥. log　dot＋拡張子だけの
ファイル名にします。
最後にログ・ファイルを使用した年月日が2007年5月30日の場合
作成されるログ・ファイル名は
c:¥logs¥2007-05-30. log

ログ・ファイル　サイズ制限:ログ・ファイルの保存容量を制限することが出来ます。
チェック有り時　:ログ・ファイル記録容量を制限します。
ログ・ファイルの容量が制限値を超えると、ファイル内の古い情報を削除します。
チェック無し時:ログ・ファイル制限はありませんので、パソコンの最大DISK容量まで使用します。
最大ログ・ファイル容量:ログ・ファイルの制限容量をK－Byte単位で指定します。
古い部分をカットする割合:ログ・ファイルの制限容量に達した後でカットするファルの割合(%)を設定出来ます。
新しい記録部分をのこし、古い記録部分側を削除します。
大きな値を設定すると、カットする割合も大きくなります。

オプション–> オプション設定–>Job　Report

Job　Reportは現在プログラマで作業している情報を集約してレポートファイルを作成する機能です。Job　Reportはプロジェクトファイルの読込みから開始して、次のプロジェクトファイルの読込まで、またはプログラマの作業終了（PG4UWソフト終了）までの集約情報を記録したファイルです。

Job　Reportには次の情報が含まれています。

- プロジェクト名
- プロジェクト年月日
- プロテクトモードステータス
- PG4UWソフトウェアのバージョン
- プログラマ名称とシリアル番号
- Job開始時間（プロジェクトファイルを読込んだ時間）
- Job終了時間（Job　Reportが作成された時間）
- デバイス名
- デバイスの種類
- チェックサム
- デバイス操作オプション
- シリアライズ情報
- デバイスのPASS・FAILカウント数

Job　Reportは次の場合に作成されます。

1. プロジェクトファイル読込みコマンドを使用した時
2. 接続していたプログラマが開放された時
3. PG4UWソフトを終了した時
4. デバイスカウントダウン設定数が0になった時（書込み予定数量が完了した時）
5. ユーザーがJob　Report操作を実行した時

Job　Report機能を有効にする

- チェック有り時：Job　Report機能が有効になります。
  上記のJob　Report作成条件が成立した時にJob　Reportが作成されます。
- チェック無し時：Job　Reportは作成されません。

Job　Reportファイルを自動保存

- チェック有り時：Job　Reportが自動的に保存されます。
  Job　Reportディレクトリボックスで指定した場所・ファイル名に保存されますが、Job　Report情報画面は表示されません。

保存されるファイル名のフォーマットは

Job Report_**連続番号**_プロジェクト名.jrp

_**連続番号**は保存ファイルにつけた連続番号です。この番号が既に存在する場合はこの番号を＋1した新しい番号のレポートファイル名で保存します。
_プロジェクト名は現在使用しているプロジェクトファイル名で、ファイル拡張子を除いた名称が使用されます。

例1：使用中のプロジェクトファイル名　c:¥myproject.eprj　保存するディレクトリ名　d:¥job_reports
この保存するディレクトリにまだJobレポートファイルが存在しない場合
作成されるJob　Reportファイル名は
d:¥job_reports¥job_report_000_myproject.jrp
となります。

例2：前記の1．と同一条件で、保存するディレクトリに既にJobレポートファイルが存在する場合
作成されるJob　Reportファイル名は
d:¥job_reports¥job_report_001_myproject.jrp
となります。
ファイル名に含まれている**連続番号**部分が＋1されています。

オプション　-> オプション設定　-> Automatic YES!

このオプションでは Automatic YES! で作業する時の、 'BUSY_LED' の点灯方法を指定します。

デバイス交換時の LED 表示

プログラマの初期設定では、デバイス書込み終了時と新しいデバイスに交換されるのを待つ時に'BUSY_LED' を点滅させます。
但しマルチプログラマではこの表示機能はありません。

BUSY_LED が点滅するモード

デバイス書込み終了時と新しいデバイスに交換されるのを待つ時に 'BUSY_LED' を点滅させます。デバイス動作が完了すると、その動作結果を示す 'GOOD_LED' または 'ERROR_LED'を点灯します。そして 'BUSY_LED' を点滅させます。
その後作業完了したデバイスがソケットから取り除かれると、 'GOOD_LED' と 'ERROR_LED' は消灯します。
しかし 'BUSY_LED' は新しいデバイスに交換されるまで点滅を続けます。
プログラマは新しいデバイスを検出（ソケットにデバイスの PIN が接触）した時点で、LED を点滅から常時点灯に切り替えます。
この後一定時間以内にデバイスが正常に実装されたことが確認されると、プログラマは動作を開始します。

LED 表示無しモード（quiet mode）

この設定ではデバイス書込み終了時と新しいデバイスに交換されるのを待つ時にも 'BUSY_LED' は表示しません。
但しデバイス動作結果を表示する 'GOOD_LED' と 'ERROR _LED' は表示します。

オプション–> オプション設定–>リモート･コントロール

他のアプリケーションからリモート･コントロール機能を使用して、プログラマがサポートしている機能を使用することが出来ます。この機能は、ハンドラなどを使用した大量書き込みシステムや、他のアプリケーションにとても有効です。
コントロールソフトを制御するリモートアプリケーションがServer（サーバー）として機能します。
コントロールソフトはClient（クライアント）として機能します。リモートアプリケーションとコントロールソフトはTCP／IPプロトコルで通信されます。
コントロールソフトを1台のパソコンにインストールし、他の1台にリモートアプリケーションをインストールし、それらをネットワークで接続します。
DefaultのTCP通信設定は　ポートアドレス　127．000．000．001またはローカルです。
リモート･コントロールのポートアドレスは127．0．0．1もしくはローカルホストになります。
ポートアドレス（Client）設定はコントロールソフトにだけ設定します。Clientパソコン（プログラマを接続したPC）とServerパソコン（リモートアプリを入れたパソコン）のポート設定をおこないます。
Default設定は1台のパソコンにリモートアプリとコントロールソフトをインストールしたローカル設定になっています。
注意：ファイアオールがインストールされている場合、サーバーもしくはクライアントがスタートする際に、警告メッセージを表示します。ファイアオールがネットワーク接続に警告を表示した場合、“Allow”を選択してください。そうしないとリモート操作は、動作しません。

オプション–> オプション設定–> オプション保存方法

コントロールソフトを終了する時にプログラマ設定を保存するかどうかを選択します。
下記の3つの選択があります。

アプリケーション終了時のオプション設定を保存

オプション設定を保存しない　：プログラマ設定を保存せずに、コントロールソフトを終了します。

オプション設定を自動保存する：プログラマ設定を保存し、コントロールソフトを終了します。

オプション設定を保存する前に確認する
：コントロールソフトを終了する前に、ユーザーがプログラマ設定を保存するか、しないか選択することが出来ます。

オプション–> オプション設定>その他

このオプション設定はコントロールソフトの処理優先度、起動時のディレクトリ、ツールバーの表示スタイル等を設定します。

アプリケーションの優先度：コントロールソフトの処理優先度を設定出来ます。
この設定は、書き込み時間の早さに影響を与えることになります。
特にON_Demandタイプのアプリケーションが動作している場合は動作が遅くなります。レベルをLOWに設定した場合書き込み時間が遅くなりますので、注意が必要です。

起動時ディレクトリ　：プログラム起動時にデータファイルのディレクトリが指定出来ます。

初期値の起動時ディレクトリを使用する
：このプログラムを起動したディレクトリが使用されます。

プログラムが終了した時のディレクトリ
：最後に使用していたディレクトリが使用されます。

ユーザーが指定する　：ディレクトリをユーザーが指定します。

ツール ボタン　:ツールボタンの表示スタイルと、ヒント表示を設定します。

ラベル スタイル　:デバイスaddress表示、PASS／FAIL数表示、デバイス構成部分の背景表示を指定します。

・プログラマ動作時の LED 色:

新タイプのプログラマでは LED 色 が変更できます。
・標準設定 LED 色（ERROR=赤色, BUSY=黄色）
・旧タイプ設定 LED 色（ERROR=黄色, BUSY=赤色）
Note:現在使用しているプログラマが **LED 色の変更** 機能をサポートしていない場合
プログラマのメニューの LED 色の変更設定は出来ません。

旧タイプのプログラマの場合、ソフトウェアで点灯する LED を変更します。
・標準設定 LED 色（ERROR=赤色, BUSY=黄色）
・旧タイプ設定 LED 色（ERROR=黄色, BUSY=赤色）
Note: 旧タイプのプログラマのみで対応しています。

オプション->ツールバー

このコマンドはツールバーの表示／非表示を指定します。
またデバイス動作前のデバイスオプション設定表示／非表示を指定できます。

オプション->ツールバー ->メインツールバー

メインツールバーの表示／非表示と表示スタイルが指定出来ます。

メインツールバー（標準スタイル）
メインツールバー（代替スタイル）
メインツールバーなし

オプション->ツールバー ->拡張ツールバー

拡張ツールバーの表示／非表示が指定出来ます。

オプション->ツールバー ->Program前にデバイスオプション表示

デバイスの書き込み／消去動作前に、デバイスオプション->動作オプション ウィンドウの表示／非表示を指定出来ます。

オプション–>操作プロテクトモード

操作プロテクトモードは特別な機能で、コントロールソフト画面上の一部操作を禁止します。

プロテクトモードが設定されている場合、プログラマの動作モード、デバイス選択、デバイスのオプション設定、バッファデータ編集、データの読み込み／保存等を禁止することが出来ます。
プロテクトモード は作業者の間違った操作を防ぐことができますので、大量のデバイスを書き込む作業に適しています。

1個書きプログラマのプロテクトモード設定

２種類のプロテクトモードがあります。

1. メニューコマンドの オプション –> 操作プロテクトモード を使用する。
   この操作によりパスワード設定用のダイアログが表示されます。
   ユーザーはパスワードを入力し、続いて確認用のパスワードを入力します。
   プログラマは2つのパスワードを確認し、一致していれば操作プロテクトモードになります。

2. プロテクトモード設定済みのプロジェクト ファイルを読み込む。
   リードしたプロジェクト内でプロテクトが設定されていますので、プロテクトが有効になります。
   詳細は “ファイルー＞プロジェクトを保存“ を参照して下さい。

パスワード入力とプロテクトモード設定画面

・プロジェクトの読込みは可能とする:
ディフォルト設定では、プロテクトモード中に 'LOAD prj' ボタンとメニュー内の'プロジェクトの読込み' は使用出来ません。
この設定を有効（チェック有り）にした場合、プロテクトモード中でも 'LOAD prj' ボタンとメニュー内の 'プロジェクトの読込み' が使用可能になります。

・バッファの表示／編集 及び Alt+S 設定を禁止する:
ディフォルト設定では、プロテクトモード中でも 'View/Edit' ボタンとバッファメニュー内の '表示/編集' 機能が使用出来ます。
バッファの内容は表示確認出来ますが、変更は出来ません。

この設定を有効（チェック有り）にした場合、プロテクトモード中は バッファデータの表示も出来ません。

・操作プロテクトモードが有効になった場合は、Log Window 上の右上部分に

Protected mode

を表示します。

プロテクトモード用オペレーションの選択

・Multi オペレーション” モード：

プロテクトモードの基本形です。デバイスリード以外のデバイス操作（Blank、Verify、Program、Erase）が全て使用できます。
このモードではリード操作が禁止されていますので、作業者が間違ってマスターデータを変更してしまう事がありません。
リード以外のデバイス操作が出来ますので、汎用的なプロジェクトとして使用出来ます。

Multi オペレーション” モード ， Read 以外のデバイス操作が可能

・One オペレーション” モード：

プロテクトモードの拡張版です。使用する **デバイス操作を 1 つだけ** に限定しています。
プログラマの動作が確定していますので、作業ミスが防止できます。同一デバイスを大量に処理する場合に適しています。

One オペレーション” モード ， 1 つのデバイス操作のみ可能

プロテクトモードの解除方法

２種類の解除方法があります。

1．メニューコマンドの **オプション -> 通常モード(プロテクト解除)** を使用する。
　　プロテクトモード設定時に入力したパスワードを入力します。
　　プロテクトモード設定時のパスワードと一致していればプロテクトモードは解除されます。

2．プロテクトモードを中止する他の方法として、コントロールソフトを終了する方法があります。
　　プロテクトモードはコントロールソフト動作中だけ有効なモードですので、終了時に解除されます。
　　コントロールソフト再起動時にはプロテクトモードは解除されています。

・他のオプション **'プログラム開始前にプロジェクトファイルID番号を入力する'** で使用する **'ID番号'** は
　メニュー内の “ファイル－＞プロジェクトを保存“ を参照して下さい。

オプション->マルチ・プロジェクト

マルチ・プロジェクトは、プログラマの動作をユーザーが自由に組み合わせて実行することが出来るようした特別の機能です。
マルチ・プロジェクト作成時や Sub・プロジェクト作成時に記録された情報をベースにして作成されます。

マルチ・プロジェクトを使用すると
・Multi-chip デバイスの書込みが可能です。
・プログラマ動作の組み合わせが可能です(例:Program + Verify + Verify + Verify)。
モードの詳しい説明はプロテクトモード時のオペレーションを参照して下さい。

マルチ・プロジェクトに関連する語句

・**マルチ・プロジェクトファイル** – 全てのマルチ・プロジェクト情報を含む特別なファイルです。
マルチ・プロジェクトファイルには1個以上のプロジェクトファイルを入れることが出来ます。
マルチ・プロジェクトに含まれるプロジェクトは **Sub プロジェクト** とも呼ばれます。

・**Sub・プロジェクト** – マルチ・プロジェクトファイル内に入れたデバイスの動作を指定したプロジェクトファイルのことです。

・**プロジェクトファイル** – 特別なファイルで、Buffer データとデバイス操作オプションといくつかの操作保護機能で構成されています。
デバイスをどのように処理するか、ここで設定します。一旦保存すれば、いつでも再読込可能で、簡単に実行出来ます。

・**Multi-chip デバイス** – 2 個以上の個別チップを 1 パッケージに収めたデバイスのことです。

・**Sub デバイス** – Multi-chip デバイス内の独立したデバイスを表します。
Sub デバイスは、プログラマのデバイス選択リストから選択します。
選択したデバイスは、デバイス毎のアルゴリズムでアクセスします。
Sub デバイス単独で、プロジェクトファイルの定義、保存、デバイスの書込みチェックをすることが出来ます。

・**Master デバイス** – Multi-chip デバイス・ユニットであり、Sub デバイスを含んでいます。
この Master デバイスも、プログラマのデバイス選択リストから選択します。
ユーザーはマルチ・プロジェクト・ウィザードを使用して、個別のプロジェクトファイルを組み合わせて、マルチ・プロジェクトファイルを作成出来ます。

マルチ・プロジェクトが 1 個のチップのみで構成されている場合、Master デバイスは指定出来ません。

・**デバイス オペレーション** – プログラマの動作はメニューから選択して直接実行することが出来ます。 これらの動作の内 Program , Erase はデバイスオプションを使用していることがあります。この設定は デバイス->デバイスオプション->オプション設定 で変更出来ます。

・**マルチ・プロジェクト・ウィザード** – マルチ・プロジェクトファイルを構築するための機能です。
ウィザードはマルチ・プロジェクトファイルに入れるプロジェクトを選択し、保存します。
マルチ・プロジェクトに記録したプロジェクト(Sub プロジェクト)に従ってデバイス書込みが行なわれます。
マルチ・プロジェクト・ウィザードの詳細は以下を参照して下さい。

オプション−>マルチ・プロジェクト・ウィザード

マルチ・プロジェクト・デバイス操作には マスターデバイスに関連した Sub プロジェクトが含まれている マルチ・プロジェクトファイルが必要です。
このマルチ・プロジェクトファイルはマルチ・プロジェクト・ウィザードで作成することが出来ます。

ウィザードは次の機能を持っています。
1. Sub プロジェクトを選択し、最終的なプロジェクトファイルを構築します。
2. 保存してあるマルチ・プロジェクトファイルを読込みます。
3. 最新のマルチ・プロジェクトのオペレーションでデバイス作業をスタートします。
Note:
マルチ・プロジェクトファイルはメインメニューの **プロジェクトの読込み** または **マルチ・プロジェクト・ウィザード** の Multi-prj の読込みコマンドが使用出来ます。

マルチ・プロジェクト・ウィザード ダイアログには次の表示／ボタンがあります。

・LOAD multi-prj ボタン
このボタンは、マルチ・プロジェクトファイル読込みに使用します。

・Build multi-prj ボタン
このボタンは、Sub プロジェクトテーブルに表示しているプロジェクトから、新規マルチ・プロジェクトファイルを構築する時に使用します。

・Table 1:
最新のマルチ・プロジェクトに保存されている Sub プロジェクトリストを表示しています。

・Add prj ボタン
Table 1 のプロジェクトリストに新規プロジェクトファイルを追加する時に使用します。

・Remove prj ボタン
Table 1 のプロジェクトリストからプロジェクトファイルを削除する時に使用します。

・Move up / Move down ボタン
Table 1 内の選択したプロジェクトファイルを上下に移動させる時に使用します。
プロジェクトはここで指定した順番に実行されます。最も上の行のプロジェクトが最初に実行されます。

・Help ボタン
この HELP を表示します。

・Blank, Verify, Program, Erase, resp.RUN ボタン
Sub プロジェクトテーブルにリストされているデバイスを、ここで選択したプログラマモードで実行します。
・Multi オペレーションモード:使用可能な全てのプログラマ動作から選択して作業することが出来ます。
・One オペレーションモード:限定した 1 つのプログラマ動作だけが指定されますので、間違った操作を防止することが出来ます。
モードの詳しい説明は **プロテクトモード時のオペレーション** を参照して下さい。

Note:
マルチプログラミングではシリアライズ機能(連続番号書込み機能)はサポートしていません。
またカウントダウン機能もサポートされていません。

マルチ・チップデバイスの書込みに、マルチ・プロジェクトを使用する

Multi-chip デバイス(または Single-chip デバイス)にマルチ・プロジェクトを使用して書込む為には、次の2つの基本作業が必要です:

1. マルチ・プロジェクト(またはマルチ・プロジェクトファイル)を作成する。
2. マルチ・プロジェクトをロードし、デバイスの書込み作業を行う。

1. マルチ・プロジェクト(またはマルチ・プロジェクトファイル)を作成する

以下のステップは推奨するマルチ・プロジェクト作成方法です:

1-a. Multi-chip 内の各 Sub デバイスに従来のプロジェクトと同様の方法で標準プロジェクトを作成します。
- Multi-chip デバイスに搭載されている Sub-デバイスを選択します。*1
- このデバイスのパラメータ等の必要な項目を設定し、書込むデータをバッファにロードします。
- 確認のために、この設定を使用してデバイスの書込み試験を行ないます。
- 全て OK の場合はプロジェクトファイルとして保存します。

1-b. マルチ・プロジェクトを使用するためには、Master-Multi-chip デバイスを選択します。
Multi-chip デバイスを選択すると、自動的に Multi-project Wizard が起動します。

1-c. Multi-project Wizard 内の**"Add prj ボタン"**を使用して、必要プロジェクトを追加します。
追加するプロジェクトは、Multi-chip デバイス の Sub デバイス対応プロジェクトです。

1-d. Sub プロジェクトの選択が完了した後で、**Build multi-prj ボタン**を押して、最終的なマルチ・プロジェクトファイルを作成します。
最終的なマルチ・プロジェクトファイルには、"Table 1:Sub-projects"に表示されている Sub プロジェクトが全て含まれています。

Note:

従来使用していた全てのプロジェクトファイルからマルチ・プロジェクトが作成可能です。 Master デバイスがいつも必要になる訳ではありません。
マルチ・プロジェクト作成時にユーザーが正しい Sub デバイスを選択しやすいようにサポートしているだけです。この機能は JTAG チェーンで、ISP プログラミングを使用して、異なったアルゴリズムで書込みを行なう場合に非常に有効です。

Multi-project Wizard は以下の操作時にオープンすることが出来ます:
・**デバイス選択**メニューから Master-Multi-chip デバイスを選択した時。
・**マルチ・プロジェクトファイル**を読込んだ時。
・メニューから**マルチ・プロジェクト・ウィザード**を指定した時

*1 デバイス選択リスト上での Master デバイスと Sub デバイスの名称の付け方:

Master デバイス:Multi-chip デバイスのオリジナル名称[パッケージタイプ]

Sub デバイス :Multi-chip デバイスのオリジナル名称[パッケージタイプ](part1)
Sub デバイス :Multi-chip デバイスのオリジナル名称[パッケージタイプ](part2)
・・・
Sub デバイス :Multi-chip デバイスのオリジナル名称[パッケージタイプ](partN)

デバイス表示例:

Master デバイス　: **TV0057A002CAGD [FBGA107]**

Sub デバイス　: **#1　TV0057A002CAGD[FBGA107] (NAND)**
Sub デバイス　: **#2　TV0057A002CAGD [FBGA107] (NOR)**

2. デバイスの書込み作業時のマルチ・プロジェクトファイルの使用方法

2-a. メニューの **project 読み込み**コマンドまたは、 Multi-project Wizard の **LOAD multi-prj** ボタンで
マルチ・プロジェクトファイルを読込みます。
マルチ・プロジェクトの読込み完了後、 Multi-project Wizard が自動的に起動します。

2-b. ウィザードは利用可能なデバイス操作(Blank, Verify, Program, Erase)のうちの1つを実行します。
通常最も利用されるのはデバイスの **Program** です。
選択されたデバイス動作は、マルチ・プロジェクトの中で定義された各 Sub デバイスの Sub プロジェクトをロードし 、Sub デバイス書込みのシーケンスとして実行されます。
このマルチ・プロジェクトが実行されると、各 Sub プロジェクト実行毎にプログレスバーが 0 にリセットされます。
Multi-chip 実行中はプログレスバーが何回も 0 に戻ったような動作になります。

2-c. 全ての Sub デバイスの書込みが終了すると、"リピートダイアログ"が表示されます。
プログラマのソケットから書込み終了したデバイスを取り、代わりに新しいデバイスを挿入します。
ダイアログ上の"Yes"ボタンかプログラマ上の"YES"ボタンを押して下さい。 Multi-chip 書込み作業が再開されます。

* もし Automatic YES! モードが設定されている場合、デバイス操作終了時には Automatic YES! ダイアログが 表示されます。
この画面にはプログラマのソケット状態と、書込み済みのデバイスの交換方法等が表示されます。
新しいデバイスをセットすると Multi-chip デバイス書込み作業が自動的に開始されます。
詳細は "Programmer -> AutomaticYES!" を参照して下さい。

Note:

マルチプログラミングではシリアライズ機能(連続番号書込み機能)はサポートしていません。
またカウントダウン機能もサポートしていません。

オプション->オプション設定を保存

このコマンドは現在設定されている保存可能な設定情報を全て保存します。

次の情報が保存されます。

- ・オプションメニューに含まれる各コマンド
- ・過去に選択した20個のデバイスコード
- ・使用ファイルの履歴
- ・メイン画面のサイズと表示位置

この操作でオプション設定情報が保存されますが、アプリケーション終了時に自動的にオプション情報を保存する方法もあります。
オプション-> オプション設定>オプション保存方法でアプリケーション終了時にオプション設定を自動的に保存する / 保存しない を選択する事が出来ます。

ヘルプ（メインメニュー・コマンド）

ヘルプメニューには、サポートデバイスとサポートプログラマの表示、使用中のプログラマの製造番号及びコントロールソフトのバージョン番号等の表示機能があります。

ヘルプ->マニュアル

コントロールソフトの操作マニュアルを表示します。
＜Ｆ１＞keyを押すと、操作マニュアルが表示されます。
メインメニュー内の項目をマウスで選択しておいてから＜Ｆ１＞keyを押すと選択している項目に関連するＯｎ　Ｌｉｎｅマニュアルが表示されます。
デバイスの動作中は＜Ｆ１＞keyは使用できません。
次のヘルプ内の項目は強調表示されます。

ショートカットなどのKey操作
重要な項目
表示しているヘルプ内容の詳細または関連する説明にリンクしている項目

プログラマ内のヘルプ（マニュアル内容）は、コントロールソフトのバージョンアップ時に最新版に更新されます。しかしこのマニュアルの記載内容は変更出来ませんので、疑問点がある場合はプログラマ内のヘルプ（マニュアル内容）を確認して下さい。

ヘルプ->マニュアル検索

操作マニュアル内で調べたい項目をキーワードで検索出来ます。

ヘルプ->対応デバイス

サポートしている全てのデバイス名を表示します。
ここで表示しているデバイスは、Ｍ１８８３以外でサポートしているデバイス名も表示しています。
Ｍ１８８３プログラマでサポートしているデバイス名はメインメニューの デバイス->デバイス選択
またはツールバーの Select で表示されます。

ヘルプ->対応プログラマ

コントロールソフトで使用可能なプログラマを表示します。

ヘルプ->デバイス リスト(使用中のプログラマ)

現在使用中のプログラマでサポートしているデバイスLISTを作成します。
コントロールソフトがインストールされているディレクトリにファイル名“XXXXDEV．txt”と
“XXXXDEV．htm”が作成されます。

ヘルプ->デバイス リスト (全プログラマ)

サポートしている全てのデバイスのデバイスLISTを作成します。
コントロールソフトがインストールされているディレクトリにファイル名"XXXXDEV. TXT"と"XXXXDEV. HTM"が作成されます。(XXXXにはプログラマ名が入ります)
注意:このデバイスLIST作成を実行すると、プログラマに設定されているDefaut Device設定情報(過去20件のデバイス使用履歴)が全て削除されます。

ヘルプ->デバイス リスト(クロス・リファレンス)

サポートしているデバイスとサポートしているプログラマのクロス・リファレンス表をデバイスメーカー別に作成します。
コントロールソフトがインストールされているディレクトリにメーカー選択ファイル"TOP__DEV. htmt"を作成し、その下のディレクトリにメーカー別のクロス・リファレンス表を作成します。

ヘルプ->障害レポートの作成

プログラマで発生したトラブルや故障状態を"Problem Report"としてファイルに記録します。このなかには自己診断結果や接続しているパソコンの基本情報、書き込みデバイスのConfigration情報等が含まれています。
トラブルがユーザー自身で解決できない場合、プログラマ本体にこのレポートを添付し、プログラマメーカーに送ってください。
プログラマの修理情報として利用されます。

ヘルプ->バージョン情報

コントロールソフトのバージョン番号、ミナトエレクトロニクスのホームページとメールアドレスを表示します。

保証規定

ここでは、M1883の製品保証について記述しています。
弊社から出荷後、1 年間の無償修理期間を設けていますが、保証の制限により、無償修理保証に該当しない場合がありますので、ご使用の前に、下記の規定内容を必ずお読みください。
ご不明な点がございましたら弊社までご連絡ください。

・無償保証期間

無償保証期間は、弊社より出荷後、1 年間とさせて頂きます。

・無償保証に該当するもの

弊社より出荷後、1 年以内の製品。
正常なご使用状態のもとで故障した場合。
M1883本体のハードウェア部分。
日本国内で使用していた故障した場合。
（This warranty is valid only in Japan）

・有償修理に該当するもの

保証期間内であっても、次の項目に該当する場合は、有償修理となります。
お客様の誤ったお取り扱いによる事故。
天災による事故。
デバイス書込み用ZIFソケット、プログラマ本体前面の ISP コネクタ。

・保証対象外

次の項目に該当することは弊社では、一切の責任及び、保証は出来ませんので、ご了承ください。
デバイス及びデータの保証。
故障時に生じた直接的、間接的な費用。
国内以外で使用される場合。
弊社純正品以外の機器を使用した際に生じた不具合及び、トラブル（他社の変換アダプタ/ソフトウェアの使用）。
弊社以外で修理や改造等を行ったもの。
取扱説明書に反する使用方法によって生じた故障。
M1883のバージョンアップ
　　対応デバイス追加等のバージョンアップは別途、費用が掛かります。

以下はM1883を最良の状態でご使用頂く為の記述事項です。

・M1883バージョンアップ(デバイス/ソフトウェアサポート)

新規デバイス対応、デバイス仕様変更によるソフトウェア等のバージョンアップを常時行なっています。M1883が無償期間内であっても、バージョンアップサポートは、保証対象外の為、これらの作業は、別途、費用が掛かりますので、ご了承ください。

・定期校正

M1883は、生産機器であり、安心してご利用頂く為にも、日頃の点検と年 1 回の定期校正を推奨します。

・修理、定期校正、バージョンアップについて

有償、無償に関わらず、出張修理は行っておりません。
また、代替え機等はご用意しておりませんのでご了承ください。

定期校正品、修理品を弊社宛へ発送して頂く際、発送に掛かる輸送費用はお客様負担とさせて頂きますのでご了承ください。
本製品が梱包されてくる梱包箱(製品箱)は修理等で輸送する時に利用出来ますので、大切に保管してください。本製品を輸送する場合は、弊社指定の梱包箱(製品箱)と梱包材を使用し、しっかりと梱包してください。
梱包が不十分ですと製品が破損する場合があります。輸送時に発生した破損部分の費用は、お客様の負担とさせて頂きますので、ご了承願います。
作業完了品をお客様へ返送させて頂く場合は、輸送費は弊社にて負担させて頂きます。

お問い合わせ先一覧

ミナトエレクトロニクス株式会社

本社営業部	〒224-0026	横浜市都筑区南山田町 4105 TEL 045-591-5605 FAX 045-592-2854
福岡営業所	〒812-0011	福岡市博多区博多駅前 3-6-12 オヌキ博多駅前ﾋﾞﾙ 4F TEL 092-475-2825 FAX 092-481-3502
大阪営業所	〒553-0003	大阪市福島区福島 5-16-15 福島宮脇ビル 2F TEL 06-6453-8911 FAX 06-6453-8912